EuroSPI 2009 Proceedings

Proceedings

The papers in this book comprise the industrial proceedings of the EuroSPI 2009 conference. They reflect the authors' opinions and, in the interests of timely dissemination, are published as presented and without change.

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

DELTA Series about Process Improvement – ISBN 978-87-7398-151-1.

EuroSPI

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

EuroSPI conferences present and discuss practical results from improvement projects in industry, focussing on the benefits gained and the criteria for success. Leading European industry are contributing to and participating in this event. This year's event is the 16th of a series of conferences to which countries across Europe and from the rest of the world contributed their lessons learned and shared their knowledge to reach the next higher level of software management professionalism.

EuroSPI Chairs

General Chair	Dr Richard Messnarz, (ISCN, Ireland)
EuroSPI Marketing Chair	Prof. Miklos Biro (Corvinus University, Hungary)
Scientific Program Chair	Prof. Juan Cuadrado Gallego (University of Alcala, Spain)
Scientific Program Chair	Dr Ricardo Rejas Muslera (University of Alcala, Spain)
Scientific Program Chair	Dr Rory O Connor (Dublin City University, Ireland)
Scientific Program Chair	Prof. Kari Smolander (Lappeenranta University of Tech. Finland)
Scientific Program Chair	Dr Nathan Baddoo (University of Hertfordshire, UK)
Industrial Program Chair	Risto Nevalainen FiSMA and STTF
Industrial Program Chair	Stephan Goericke, ISQI
Industrial Program Chair	Jorn Johansen, DELTA
Industrial Program Chair	Mads Christiansen, DELTA
Industrial Program Chair	Nils Brede Moe, SINTEF
Tutorial Chair	Dr Richard Messnarz, ISCN
Organizing Chair	Adrienne Clarke BA, ISCN

Industrial Program Committee

BACHMANN Volker SIBAC GmbH GERMANY

BALSTRUP Bo Center for Software Innovation DENMARK

BARAFORT Beatrix Centre de Recherche Public Henri Tudor LUXEMBOURG

BRESKE Eva Robert Bosch GmbH GERMANY

CHRISTIANSEN Mads DELTA DENMARK

COLLINO Alessandro ONION S.p.A. ITALY

COLOMO PALACIOS Ricardo Universidad Carlos III de Madrid SPAIN

DAUGHTREY Taz H. James Madison University USA

DAVIES Richard Schlumberger WesternGeco NORWAY

DUSSA-ZIEGER Claudia Method Park Software AG GERMANY

EKERT Damjan ISCN AUSTRIA

FEHRER Detlef SICK AG GERMANY

HAGENMEYER Philipp ZF Friedrichshafen AG GERMANY

HIND Tim Axa Sun Life UK

JOHANSSON Mika Finnish Software Measurement Association FiSMA FINLAND

KASTNER Norbert sepp.med gmbh GERMANY

LICHTENECKER Gerhard MAGNA STEYR Fahrzeugtechnik AUSTRIA

MESSNARZ Richard ISCN IRELAND

MORGENSTERN Jens TUEV GERMANY

NEVALAINEN Risto Falcon Leader Oy FINLAND

O LEARY Eugene EQN Ltd. IRELAND

PIKKARAINEN Mika VTT Technical Research Centre of Finland FINLAND

POTH Alexander SQS Software Quality Systems AG GERMANY

RENAULT Samuel Centre de Recherche Public Henri Tudor LUXEMBOURG

RIEL Andreas Grenoble Institute of Technology FRANCE

ROMCEA Cristina Conti Temic microelectronic GmbH GERMANY

SANDERS Marty UK

SCHWEIGERT Tomas SQS Software Quality Systems AG GERMANY

SEHL Georg iMBUS AG GERMANY

SMITE Darja Blekinge Inst. of Technology SWEDEN

SPORK Gunther MAGNA Powertrain AUSTRIA

STEFANOVA-PAVLOVA Maria Center for Innovation and Technology Transfer-Global BULGARIA

ULSUND Tor Geomatikk AS NORWAY

VON BRONK Peter Systemberatung Software-Qualität GERMANY

WÖRAN Bruno DANUBE AUSTRIA

EuroSPI Board Members

Editors of the Proceedings

Richard Messnarz, ISCN, Ireland

Mads Christiansen, DELTA, Denmark

Jorn Johansen, DELTA, Denmark

Sonja Koinig, ISCN, Austria

Juan Cuadrado Gallego, University of Alcala, Spain

Welcome Address by the EuroSPI General Chair

Dr Richard Messnarz

EuroSPI is an initiative with 4 major goals (www.eurospi.net):

1. An annual EuroSPI conference supported by Software Process Improvement Networks from different EU countries.
2. Establishing an Internet based knowledge library, newsletters, and a set of proceedings and recommended books.
3. Establishing an effective team of national representatives (in future from each EU country) growing step by step into more countries of Europe.
4. Establishing a European Qualification Framework for a pool of professions related with SPI and management. This is supported by European certificates, exam systems, and online training platforms (www.ecqa.org).

EuroSPI is a partnership of large Scandinavian research companies and experience networks (SINTEF, DELTA, STTF), the ASQF as a large German quality association, the American Society for Quality, and ISCN as the co-coordinating partner. EuroSPI collaborates with a large number of SPINs (Software Process Improvement Network) in Europe.

EuroSPI conferences present and discuss results from software process improvement (SPI) projects in industry and research, focusing on the benefits gained and the criteria for success. Leading European universities, research centers, and industry are contributing to and participating in this event. This year's event is the 16th of a series of conferences to which international researchers contribute their lessons learned and share their knowledge as they work towards the next higher level of software management professionalism.

The greatest value of EuroSPI lies in its function as a European knowledge and experience exchange mechanism for Software Process Improvement and Innovation of successful software product and service development. EuroSPI aims at forming an exciting forum where researchers, industrial managers and professionals meet to exchange experiences and ideas and fertilize the grounds for new developments and improvements.

EuroSPI also established an umbrella initiative for establishing a European Qualification Network in which different SPINs and national initiatives join mutually beneficial collaborations (EU Certificates Campus www.eu-certificates.org, European Certification and Qualification Association www.ecqa.org).

A cluster of European projects (supporting ECQA and EuroSPI) contribute knowledge to the initiative, including currently EU Cert (EU Certificates Campus), iDesigner (integrated mechatronics designer), MONTIFIC (Financial SPICE Assessor), ELM (e-learning manager), CROMEU (EU Managers in South eastern Europe), etc. A pool of more than 20 qualifications has been set up.

Please join the community of cross company learning of good practices!

Welcome to Spain by Prof. Juan Cuadrado Gallego and Dr Ricardo Rejas Muslera

Prof. Juan Cuadrado Gallego, University of Alcala, Spain

Dr Ricardo Rejas Muslera, University of Alcala, Spain

The University of Alcalá (Spanish: Universidad de Alcalá) is a public university located in the city of Alcalá de Henares, to the east of Madrid in Spain. Founded in 1499, it was moved in 1836 to Madrid. In 1977, the University was reopened in its same historical buildings. The University of Alcalá is especially renowned in the Spanish-speaking world as it presents each year the highly prestigious Cervantes Prize.

Today's University of Alcalá preserves its traditional humanities faculties, a testimony to the university's special efforts, past and present, to promote and diffuse the Spanish language through both its studies and the Cervantes Prize, which is awarded annually by the King and Queen of Spain in the Paraninfo (Great Hall). The University has added to its time-honoured education in the humanities and social sciences new degree subjects in scientific fields such as health sciences or engineering, spread out across its different sites (the Alcalá Campus, El Encín, and Guadalajara), all of which, together with the Science and Technology Park, are a key factor in its projection abroad, while also acting as a dynamo for activities in its local region.

Prof. Juan Cuadrado Gallego is a well known expert in the field of measurement and mathematical models used to measure processes and products. He has opublished numerous scientific papers in this field.
Dr Ricardo Rejas Muslera has done extensive research work in the use of assessment and improvement models for legal aspects, so called legal assurance processes.

We are welcoming EuroSPI 2009 to Alcala, a historic place with one of the oldest universities from Spain. University of Alacala is a member of international research networks and actively contributes process and product measurement knowledge to the European SPI community. We believe that this joint conference will fertilize the grounds for empowering networks and partnerships in SPI between the Spanish and the European wide communities.

Please join our Spanish SPI community and lets create connections Europe wide!

Welcome by DELTA, Editors of the DELTA Improvement Series

DELTA has been working with Software Process Improvement (SPI) for more than 15 years including maturity assessment according to BOOTSTRAP, SPICE and CMMI. DELTA has also been a partner in the EuroSPI conference from the very beginning 16 years ago. We are now for the 2.nd. time the publisher of the Industrial Proceedings from EuroSPI making it part of the DELTA series about Process Improvement.

**Jørn Johansen,
DELTA, Denmark**

Jørn Johansen is Manager of the DELTA Axiom department at DELTA. He has an M.Sc.E.E. from Ålborg University and more than 28 years experience in IT. He has worked in a Danish company with embedded and application software as a Developer and Project Manager for 15 years. Mr. Johansen has been involved in all aspects of software development: specification, analysis, design, coding, and quality assurance. Furthermore he has been involved in the company's implementation of an ISO 9001 Quality System and was educated to and functioned as Internal Auditor.

For the last 14 years he has worked at DELTA as a consultant and registered BOOTSTRAP, ISO 15504 Lead Assessor, CMMI Assessor and *ImprovAbility*™ Assessor. He has participated in more than 40 assessments in Denmark and abroad for companies of all sizes. He was the Project Manager in the Danish Centre for Software Process Improvement project, a more than 25 person-year SPI project and Talent@IT, a 26 person-year project that involves 4 companies as well as the IT University in Copenhagen and DELTA. Currently Mr. Johansen is the Project Manager of SourceIT an 18 person-year project focusing on outsourcing and maturity. Mr. Johansen is also the co-ordinator of a Danish knowledge exchange group: Improving the Software Development Process, which is the Danish SPIN-group.

Contact: Jørn Johansen, DELTA, Venlighedsvej 4, DK 2970 Hørsholm, Denmark, phone +45 72 19 40 00 or e-mail : joj@delta.dk

**Mads Christiansen,
DELTA, Denmark**

Mads Christiansen has an M.Sc.E.E. from DTU (Danish Technical University) and more than 30 years experience in product development and IT. He has worked for 19 years in a Danish company with embedded and application software as a Developer and Project Manager. Mr. Christiansen has been involved in all aspects of software development: specification, analysis, design, coding, and quality assurance and managing outsourced projects in Denmark and USA.

For the last 11 years he has worked at DELTA as a consultant in SPI (requirements specification, test, design of usable products and development models). Currently Mr. Christiansen works with eBusiness and as Innovation Agent. Mr. Christiansen is also *ImprovAbility*™ Assessor and Trainer of *ImprovAbility*™ project Assessors.

Contact: Mads Christiansen, DELTA, Venlighedsvej 4, DK 2970 Hørsholm, Denmark, phone +45 72 19 40 00 or e-mail : mc@delta.dk

Contents

Experience Session 1: SPI and Safety Engineering

Experience Session 2: SPI and Systems Engineering

Experience Session 3: SPI and Knowledge Engineering

Experience Session 4: SPI Implementation Experiences

Experience Session 5: SPI and Testing

Experience Session 6: SPI and Assessment Models

Experience Session 7: SPI and Measurement

Experience Session 8: SPI and Agile

Experience Session 9: Utilising Improvement Models

Experience Session 10: SPI and IT Services

Experience Session 11: SPI and SW Engineering

Experience Session 12: SPI and Processes

Experience Session 13: SPI in SMEs

Additional Requirements for Process Assessment in Safety-Critical Software and Systems Domain

Mika Johansson [1], Risto Nevalainen[2]

[1] Tampere University, Pori Unit, Finland
and FiSMA ry, Finland
[2] FiSMA ry, Finland

Abstract

Certification of safety-critical software is a multi-disciplinary topic. Process assessment is an essential part of that, but is not enough for software certification. Certification employs also several other method families, like inspections and reviews, independent V&V, conformance with selected reference standard(s) and use of selected measurements and analyses.

Process assessment supports directly qualification of safety-critical applications but is less relevant for certification of platforms and environments. Anyway, qualification and certification are closely related, because certification as a whole supports qualification and makes it more effective. It is possible to adapt and evolve process assessment so, that it supports both qualification and certification.

Typical process assessment is done for improvement purpose. In qualification and certification that is not so relevant as conformance and management of risks. In this paper we discuss about possibilities to develop process assessment to achieve that goal. In most cases assessment is a combination of several approaches.

Keywords

Safety-critical software, process assessment, conformance with standards

1 Introduction

"National Nuclear Power Plant Safety Research 2007-2010, SAFIR2010" is a Finnish four-year research programme. The objective of the programme is "to develop and maintain the nuclear safety expertise and deterministic and probabilistic methods to assess safety so that new matters related to nuclear safety appearing their significance can be assessed without delay" (SAFIR2010 2006). The planning period for national research on nuclear power plant safety up to 2010 contains granting licenses for the four power plants in use and that under construction. Know-how developed in publicly funded research programmes can be applied in licensing processes.

A research area of automation and control room in the research programme includes three on-going research projects, one of which is the "Certification facilities for software, CERFAS". The main purpose of CERFAS is to develop facilities for a consortium called Software Certification Service. Conditions for services of consortium are the application of diverse expertise and effective evaluation tools. This leads into networking both in the project and in the certification services.

Certification can be based both on generic sets of criteria and domain specific requirements. Our goal is to combine these two approaches. Most important nuclear specific requirements are standards, which include requirements for safety critical systems and software. The most relevant is IEC60880, which can be used also directly as a reference for certificate. Qualification and licensing of safety class 2 I&C systems includes a conformance statement against IEC 60880, and that is already a kind of certificate. The other main reference is IEC 61513, which is based on generic IEC 61508 (functional safety). [Harju2008]

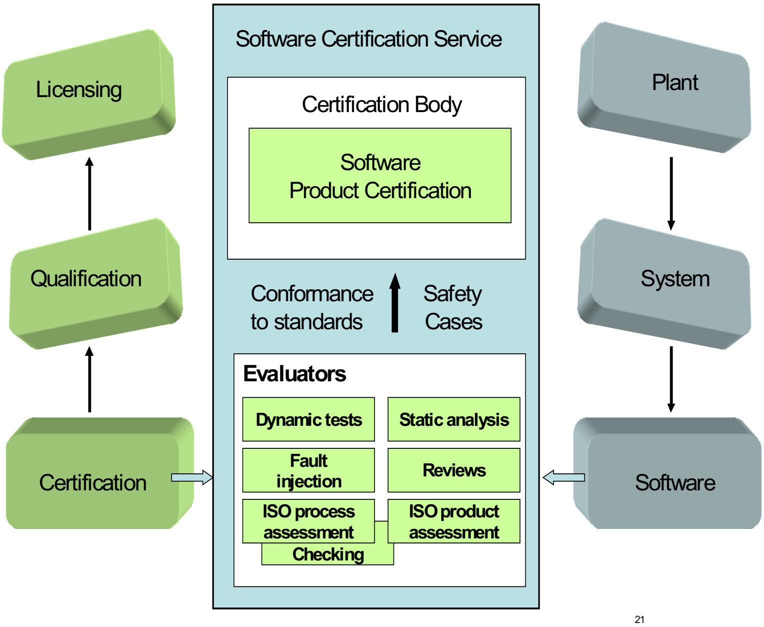


Figure 1: Main areas of topics in CERFAS project [Harju2008]

In process assessment we apply ISO/IEC15504 in quite similar way as in several other domains (automotive, medical devices, space for example). Anyway, process assessment does not work in isolation and is not enough as such. It needs to be integrated with several other approaches in software certification, as safety cases, conformance assessment and software measurement. These topics are covered in chapters 3 – 5 of this article. The concept is illustrated in Figure 2, showing some typical process assessment related topics.

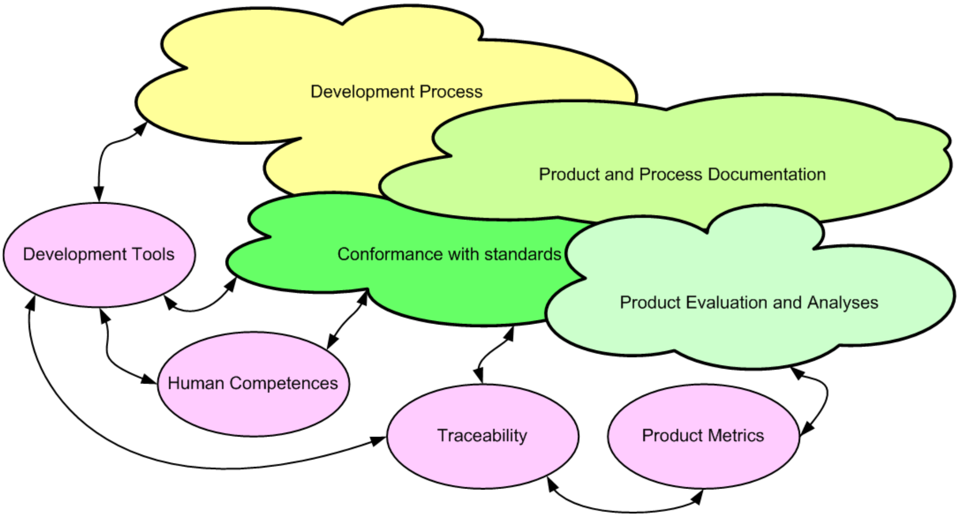

Development Process
Product and Process Documentation
Development Tools
Conformance with standards
Product Evaluation and Analyses
Human Competences
Traceability
Product Metrics

Figure 2: Process assessment related topics in safety-critical software domain.

Figure 2 contains two types of topics. Some of them (clouds in figure 2) are heavily interconnected and are always part of process assessment, more or less. For example, development process defines directly what are the most important processes in assessment scope. Again, it defines what is the most essential documentation. It leads to product evaluation. Conformance with standards is always in core of certification, because certification is based on some defined reference.

Some topics are more focused (some typical ones are shown as circles in figure 2). They are sometimes mandatory elements in certification, like proper validation of development tools. Some others are more judgement-based, like human competences and their role in developing and validating safety-critical software. Traceability and product metrics are examples of topics, which have high relevance as evidences for safety case. Note that only some relationships (arrows) are presented in figure 2, mainly for illustration purpose.

2 Additional requirements in process assessment

2.1 Basic types of assessment

In CERFAS, we have specified three different basic types and “use cases” of process assessment (see figure 3). They are needed typically as a sequence:

- Short “ability assessment” to check overall readiness to develop and deliver safety-critical software. If overall ability of software organisation is low, then it leads to cancellation of the certification process or additional time to restart it.

- Full-scale "certification assessment" to support preliminary software qualification and provide evidence for software assurance and safety case during software certification process.
- "Gap fulfilment assessment" to prevent and fix potential causes of non-conformances of products and processes and their related risks when identified during certification process.

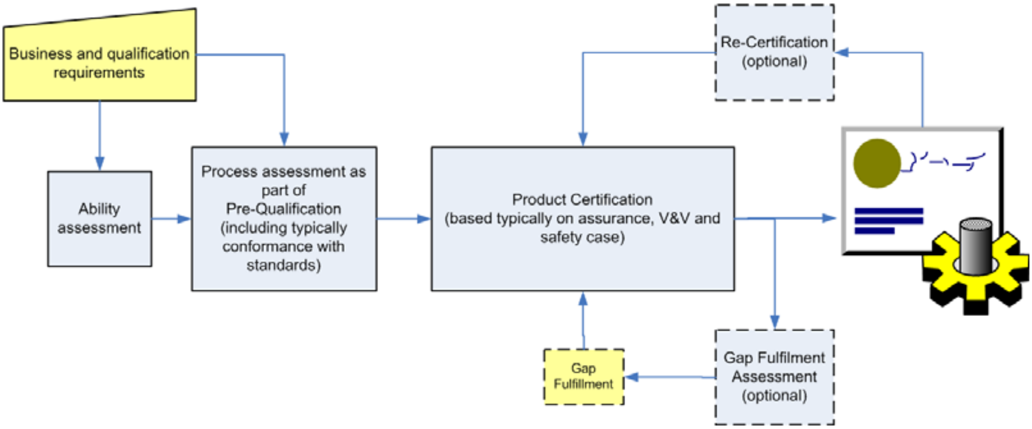

Business and qualification requirements
Ability assessment
Process assessment as part of Pre-Qualification (including typically conformance with standards)
Product Certification (based typically on assurance, V&V and safety case)
Re-Certification (optional)
Gap Fulfillment
Gap Fulfilment Assessment (optional)

Figure 3. Typical sequence of different process assessments during quaification and certification of safety-critical software

2.2 Ability assessment

Ability assessment is typically quite short, even only some days of effort. It can vary a lot, depending on the current level of software organisation and its products. Typical examples are:

- Assessment of software development processes (mainly ENG category in ISO/IEC 15504 Part 5).
- Review of core documentation, or documents from a chosen specific topic, as evidences of process capability and conformance with selected reference standard(s).
- Conformance with selected reference standard(s), for example IEC61508 Part 3 or IEC60880.

Quite often ability assessment is also a combination of several topics. In Figure 2 we presented some of them. To avoid heaviness and complexity of ability assessment, typical combination is only with two topics. An example could be conformance check + current implementation of bi-directional traceability.

2.3 Process assessment

Process assessment in CERFAS context is quite normal, SPICE – type process. Of course, it is more formal than most improvement oriented assessments. Evidences are collected and recorded systematically, and they are a solid basis for data collection, validation and ratings. Rigour of assessment is near to Scampi-A method in strictness and formalism [ARC1.2]. Results are reported as gaps to target level. Each gap can be classified by magnitude and risk, as defined in [ISO/IEC15504-4].

One additional stakeholder in process assessment is the certification body. Typical responsiblity is that customer organisation orders certification from a certification body. They decide together which references and methods is used in certification. One basic requirement is independent team for process assessment. Each team member has to fulfil competence requirements. Stakeholders and their relationships in qualification/certification driven process assessment are presented in figure 4.

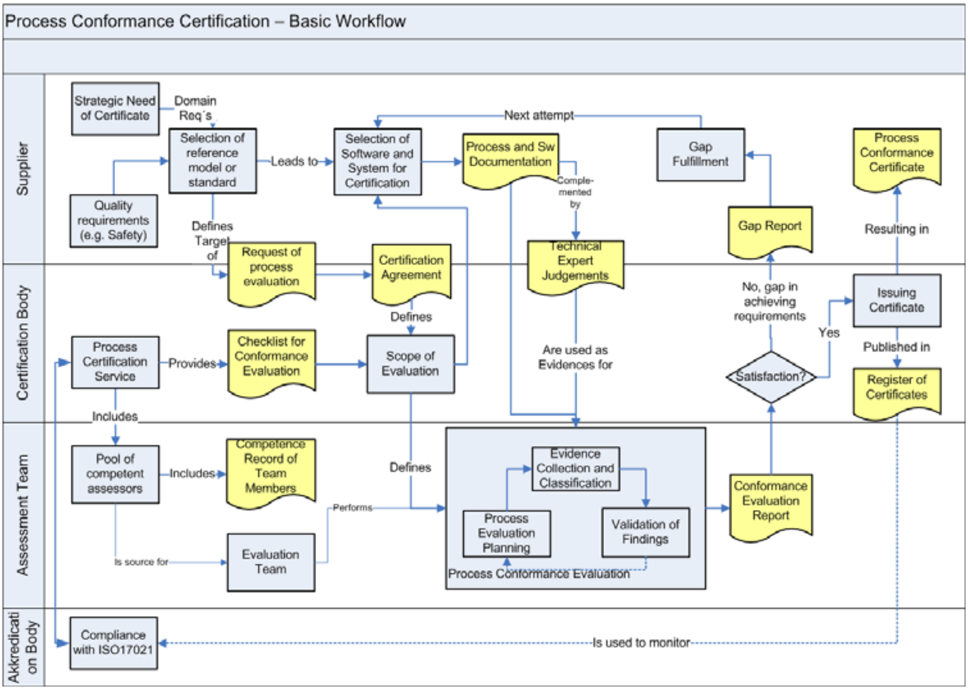

Process Conformance Certification – Basic Workflow
Supplier
Certification Body
Assessment Team
Akkreditation Body
Strategic Need of Certificate
Domain Req´s
Selection of reference model or standard
Leads to
Selection of Software and System for Certification
Next attempt
Process and Sw Documentation
Complemented by
Gap Fulfillment
Process Conformance Certificate
Quality requirements (e.g. Safety)
Defines Target of
Request of process evaluation
Certification Agreement
Technical Expert Judgements
Gap Report
Resulting in
Defines
No, gap in achieving requirements
Issuing Certificate
Yes
Process Certification Service
Provides
Checklist for Conformance Evaluation
Scope of Evaluation
Are used as Evidences for
Satisfaction?
Published in
Register of Certificates
Includes
Pool of competent assessors
Includes
Competence Record of Team Members
Defines
Evidence Collection and Classification
Conformance Evaluation Report
Performs
Process Evaluation Planning
Validation of Findings
Is source for
Evaluation Team
Process Conformance Evaluation
Compliance with ISO17021
Is used to monitor

Figure 4. Stakeholders, their main activities and typical workflow in qualification/certification oriented process assessment and conformance evaluation

One other additional requirement is satisfaction of accreditation rules. They are defined in ISO17020 family of standards. Most requirements for process assessment are same as for management system standards (for example ISO9001). Assessment process must be documented and include competence requirements. Assessment must contain audit trail between assessment phases and intermediate results. Finally, if assessment leads to process certificate, it must be publicly available for intended audience.

Most of accreditation requirements are built in process assessment standards and models. Both SPICE and CMMI model families have such guidance.

2.4 Gap fulfilment assessment

Third basic type of process assessment in CERFAS context is check of process improvements needed to get product certificate. This is needed in such cases that software is incomplete or erroneous during any phase of certification. Typical example could be design errors found during independent tests. Then the software organisation needs to change specification and/or design process so that errors can be prevented in advance or detected during design phase. Typical process improvement would be better inspection or quality assurance during early phases of software lifecycle. Sometimes also more formal process would be needed, maybe with model checking type quality assurance. These changes in development process must be verified, and one easy and straightforward way is focused process assessment. There is nothing specific compared to normal SPICE – type process assessment in this phase.

3 Additional requirements in models and references

3.1 Standards for safety-critical software and systems

In CERFAS context the primary standard for safety-critical software is IEC 60880, software aspects of computer-based systems performing category A functions, or in Finnish legislation, safety class 2 (the highest one were digital systems are allowed). IEC 60880 sets requirements and also recommendations for the development processes. Each phase of the lifecycle is addressed. The largest amount of requirements is in software design and construction topics. In the latter there are special clauses for tools and 3rd party software components like COTS.

Although many requirements of IEC 60880 are for processes, the standard claims that the most important aspects of software safety are two product-related technical features or design principles, namely self-supervision and avoidance of common cause failures. The annexes have very detailed design and programming requirements for the software product, from architectural approach to the memory usage.

The other standard used is IEC 61508, functional safety of electronic systems. The part 3 addresses software, while part 1 defines general requirements for functional safety management system. As in IEC 60880, the software lifecycle has been divided into phases, and there are specific requirements for each phase.

Although IEC 61508 claims that it is for safety related systems, some of the requirements are more strict than in safety-critical IEC 60880. For example, competences must be managed in more details, verifications are stricter and there must be independent audits during the project.

Neither of these standards have the concept of process capability nor maturity. Safety integrity levels in IEC 65108 are not comparable on process capability levels by any means. Requirements can only be divided into mandatory and optional ones, and they are either fulfilled or not. [IEC60880] [IEC61508-1] [IEC61508-3]

There are lots of other standards, from IEC 61508 based domain specific standards to many military models, but these have not been included in CERFAS.

3.2 Additional processes in generic PAM models

The software lifecycle phases in IEC 60880 and in IEC 61508 are rather basic ones, and they can be easily mapped to ISO/IEC 15504-5 processes and process' indicators. These standards include are also some requirements, where existing ISO/IEC 15504-5 processes are too open for interpretation, or are non-existent.

In the Table 1, most relevant ISO/IEC 15504-5 processes are listed, and they are mapped with most suitable chapter(s) in safety standards [Halminen 2007]. In the end of the table there are some requirements, which are not addressed in detailed enough way in part 5. These new process areas are software security, pre-developed software (PDS), development tools and safety life cycle management [Johansson 2009].

PDS and tools could be assessed using existing SUP.6 Product evaluation, but the process doesn't explicitly bring up multiple phases of the evaluation process, nor very important aspect in safety development, analysing and collecting the operational (usage) history data. Safety lifecycle management could be assessed using MAN.3 Project management with many additional notes, but since it is one of the key processes, it is better have own process for it. This process can also be used as one option for the ability assessment. Security is not addressed in part 5 at all. In practice, the content of the process is about requirements management, but it seems that there are benefits in separating safety or security requirements management from normal requirements. The last row, functional safety assessment, has been left out in CERFAS context, since the assessment is (typically) performed by an independent organisation. [Johansson2009].

Interesting finding is that measurement is not not a separate topic in safety standards.Many kinds of analyses are required, and they require lots of collected data. In this topic safety standards belong to previous generation and need to be updated. ISO/IEC 25000 standard may help in this, because it provides a quality model and a set of measures for software and systems. Safety is one topic there, but not yet well covered.

ISO/IEC 15504-5 Process	Process Name	IEC 60880 ref.	IEC 61508-3 ref.
ENG.1	Requirements elicitation		In IEC 61508-1
ENG.2	System requirements analysis		In IEC 61508-1 and – IEC 61508-2
ENG.3	System Architecture design		In IEC61508-2
ENG.4	Software requirements analysis	6.1, Annex A	7.2
ENG.5	Software design	7, Annex A, Annex B	7.4.2 7.4.3 7.4.5
ENG.6	Software construction	7, Annex B	7.4.2 7.4.6 7.4.7
ENG.7	Software integration	7	7.4.8
ENG.8	Software testing	8	7.4.8
ENG.9	System integration	9	7.5
ENG.10	System testing	9.3	7.3 7.7
ENG.11	Software installation	12.1	
SUP:1	Quality assurance	5.5	7.1
SUP.2	Verification	8	7.9
SUP.3	Validation	10	7.7
SUP.7	Documentation	Annex F	5
SUP.8	Configuration management	5.6	6.2.3
SUP.9	Problem resolution management	9.4 10.4	(7.8)
SUP.10	Change request management	11	7.8
MAN.3	Project management	5.3 5.4	7.1
MAN.4	Quality management	5.5	6
MAN.5	Risk management		61508-1
MAN.6	Measurement	(some metrics in Annex B)	
New process	Software security	5.7	(weakly in IEC 61508-1)
New process	Development tools	7.2 14	7.4.4
New process	Pre-developed software	15	Annex D
New process	Safety lifecycle management	5 (partly)	7.1.2 IEC 61508-1
NA	Functional safety assessment		8

Table 1: Mapping between 15504-5, IEC 60880 and IEC 61508 [Johansson2009]

4 Additional requirements in assessment results

4.1 Conformance vs capability results

Conformance result is typically flat and linear, based more or less Yes/No scale. Each non-conformance is recorded separately and the evaluation result consists mainly from those findings. So, it is a kind of list of negative findings. This kind of evaluation is typical when an organisation is evaluated against some requirement standard. Typical example would be ISO9001 or ISO20000 audit.

Process assessment result is much more structural. It consists from a list of processes, and each of them has a number showing the achieved level. Additionally, each process may have more technical results, like process attribute ratings. Rating scale is more continuous, as a minimum 4-point NPLF scale. SPICE- and CMMI-models have also an additional presentation, organisational maturity.

Safety standards, like generic IEC61508 and nuclear IEC60880, are conformance oriented. They may have and/or require also advanced calculations and analyses, for example to evaluate reliability of software and system. IEC61508 is done around concept of safety integrity level, and typical SIL is 3 or 4 for safety critical systems. It employs typically a set of corrective and preventive actions to achieve required reliability.

The regulatory authorities primarily need conformance results, to ensure that the system fulfils legal requirements. Finnish regulator in nuclear power industry (STUK) has its own requirements for licensing of safety-critical systems, called YVL5.5. It has requirements for technical processes and V&V, and they can be covered by conformance models. But YVL5.5 has requirements also for (quality) management system in the supplier organisation. That leads to some kind of combined multi-model solution, to cover all YVL5.5 requirements.

The licensee organisation is interested, in addition to conformance results, in the basic process capability results with risk analysis based on gaps. That can be done in several ways, but capability determination mode of ISO/IEC15504 is one specific and sophisticated model for that.

Combination of conformance and capability result is possible to achieve by classifying each evidence in both model types. Also models must be mutually mapped to cover all requirements. One problem in mapping is, that generic requirements are often more abstract and then more open for various interpretations. They are often more strict, at least when taken literally.

4.2 Assessment results and safety case

In many industry areas, including nuclear industry, the safety of the system is documented in one or more safety cases. Bishop et al. define safety case as “A documented body of evidence that provides a convincing and valid argument that a system is adequately safe for a given application in a given environment”. [Bishop1998] One of the key characteristics common to safety case and process assessment is that they both rely on objective evidences. Typically these evidences are more or less the same ones, but assessment and safety case might look after different aspects from the evidence. For example for code review report, process assessment module might see that the review is done according to process, software measurement module calculates the total coverage of code review and module testing, and competence module checks if the reviewers have had appropriate skills for the task, as shown in Figure 5.

Assessment results as such (full or partial result sets, or risk analysis based on the gaps) can also be used as one evidence in safety case, claiming that system is (or is not) programmed properly, and thus increase the confidence that the overall system is (or is not) safe. For example, one might be more confident on the quality of the end product, if engineering processes are at capability level 3 rather than 1.

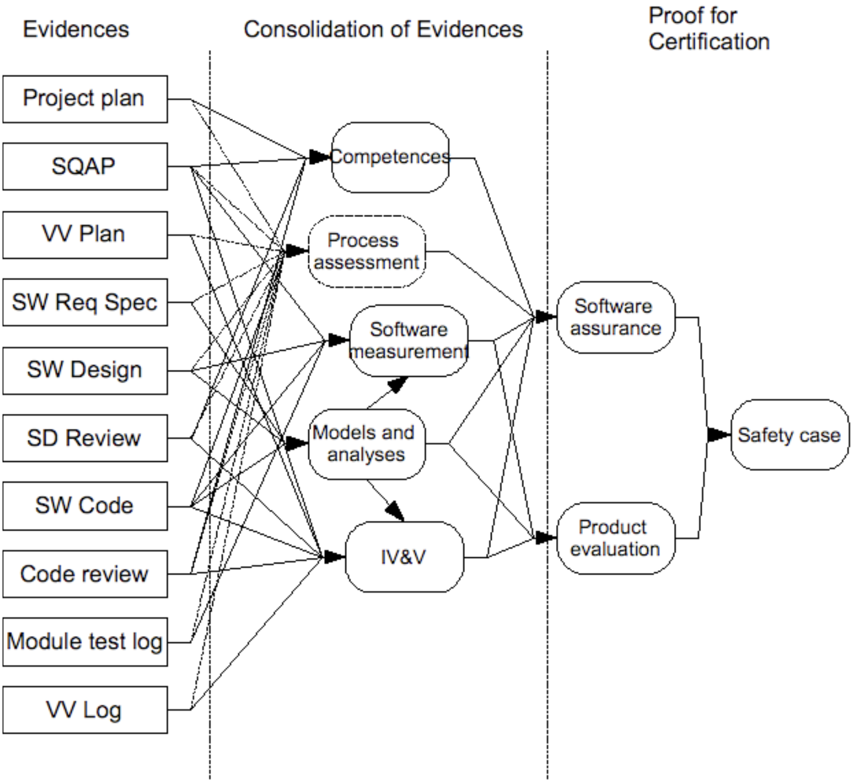


Figure 5: Same evidences are consolidated into different modules for certification. The final claim that system is (or is not) safe consists of one or more safety cases.

How the actual consolidation is done is still in a conceptual phase. Any of the standards do not give detailed requirements. For example, they can require a certain metric to be collected, but the target values of the metrics are never defined. Also, the modules in figure 5 could be arranged and linked in many ways, for example so that the “final result” would be Software Assurance Case.

5 Conclusions

This article has introduced some aspects of process assessment when it is used as part of software certification. The concepts have been piloted in the field, and they will be tested further in the next year. CERFAS project will be finished in the end of 2010, and the Software Certification Service is immidiately launched based on the project research results. The drivers for the project have been both legal and economical. The Finnish Nuclear Power Guide requires certified software in the highest safety class, so there is a need to develop a national certification scheme and certification service. Certification also shortens the actual qualification and licesing process, when the software based systems are deployed in the plant. Currently the qualification process is very costly, so all the methods and models which can support it can mean a huge savings to plant operators.

Process assessment, according our current knowledge, provides interesting insights in the safety aspects of a software product. For example, if there are gaps found in the relatively light-wight assessment, heavier methods like model checking can then focus on those weaknesses trying to find if they are endangering the actual safety. Also the ability assessment fulfills one of the industry needs, since a well documented method to get a the first go/nogo decision in purchasing process saves resources at later stages. Still, the process assessment is only a complementary method when the final validity of the product is analysed.

6 Literature

[ARC1.2] SCAMPI Upgrade Team, AppraisaCMMI®, Version 1.2., SEI report CMU/SEI-2006-TR-011, 2006

[Bishop1998] P. Bishop and R. Bloomfield, „Methodology for Safety Case Development“, In the proceedings of the Sixth Safety-critical Systems Symposium, Birmingham 1998, pp. 194-203.

[Harju2008] H. Harju, J. Lahtinen, J. Ranta, M. Johansson, R. Nevalainen , “CERFAS Summary Report”, The Finnish Research Programme on Nuclear Power Plant Safety 2007-2010 Interim report, VTT Research Notes 2466, 2008, VTT, pp. 71-83.

[Halminen2007] J. Halminen and R Nevalainen, “Qualification of Safety-critical Systems in TVO Nuclear Power Plants”, Software Process Improvement Practice, Wiley Interscience, online, 4th July 2007, pp. 559-567.

[ISO/IEC15504-4] ISO/IEC 15504 - Part 4: Information Technology - Process Assessment - Part 4: Guidance on use for Process Improvement and Process Capability Determination, 2005

[ISO/IEC15504-5] ISO/IEC 15504-5 Information Technology - Process Assessment - Part 5: An Exemplar Process Assessment Model, 2006

[IEC61508-1] IEC, IEC 61508-1: Functional safety of electrical/electronic/programmable electronic safety-related systems Part 1: General requirements, CDV version, 2008

[IEC61508-3] IEC, IEC 61508-3: Functional safety of electrical/electronic/programmable electronic safety-related systems Part 3: Software requirements, CDV version, 2008

[IEC60880] IEC, IEC 60880: Nuclear power plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category A functions, IEC, 2008

[Johansson2009] M. Johansson, “An Extended Process Set for Safety Related and Safety Critical Software Process Assessment “, In the proceedings of SPICE 2009, pp. 136-141

[Nevalainen2008] R. Nevalainen, M. Johansson, “Software process and product quality evaluation - state of the art -literature summary “, Tampere University of Technology, Research Report 5, ISSN 1795-2174, 2008

[YVL5.5] Guide YVL 5.5, Instrumentation systems and components at nuclear facilities. STUK, 2002.

7 Author CVs

Mika Johansson

Executive Director in FiSMA, partner and consultant in Spinet Oy, and part-time researcher in Tampere Technical University, Pori unit.

Mr Mika Johansson (M.Sc (Eng)) has been working as a trainer and consultant in quality and project management areas for more than ten years. He has been Executive Director in Finnish Software Measu-rement Association FiSMA since 2006. In Spinet Oy, Mika has performed assessments and audits using SPICE, CMMI, ISO9001, ISO/IEC 20000 and safety specific models since 2003.

Risto Nevalainen

Current position as a senior researcher in Tampere Technical University, Pori unit (part-time). Senior Advisor of Finnish Software Measurement Association FiSMA in software standardization and process improvement topics.

Mr. Risto Nevalainen (Lic. Tech.) has long experience in software measurement and quality topics. Nevalainen has been managing director of STTF Oy since 1996. His working experience includes also position as managing director of Finnish Information Technology Development Center during 1989-1995. Before that he had different research and management positions for example in Technical Research Centre (VTT), Technical University of Helsinki (HUT), Finnish Prime Minister's Office and Finnish Economic Planning Centre. Mr. Nevalainen has participated in ISO15504 (SPICE) standard development since beginning. He is Competent SPICE Assessor since 1996 and ISO9000 Lead Assessor since 1991.

Representation of Quality Attribute Techniques Using SPEM and EPF Composer

Yin Kia Chiam[1,2,3], Mark Staples[1,2], Liming Zhu[1,2]
[1]NICTA, Locked Bag 9013, Alexandria NSW 1435, Australia
{yinkia.chiam, liming.zhu, mark.staples}@nicta.com.au
[2]School of Computer Science and Engineering, K17, University of New South Wales, Sydney, NSW 2052, Australia
[3]School of Computer Science, University of Malaya, 50603 Kuala Lumpur, Malaysia

Abstract

There are many development techniques used to assist development teams to achieve required levels of product quality such as safety, security and performance. These ``Quality Attribute Techniques'' (QAT) aim to identify, eliminate, reduce, control and minimise potential quality problems in the development of critical systems. Although widely used, these techniques are not normally well represented in software process models. This paper proposes two alternative representations of Quality Attribute Techniques using the SPEM metamodel and Eclipse Process Framework (EPF) Composer and shows how these techniques can be incorporated into software development process models. Safety techniques have been selected as a case example for evaluation. The evaluation identifies advantages and limitations of the SPEM and EPF Composer in terms of their ability to support representation and integration of Quality Attribute Techniques. Some improvements to SPEM and EPF Composer are suggested.

Keywords

Quality Attribute Techniques, SPEM, EPF Composer

1 Introduction

The quality of software products need to be developed and assured throughout the development process [1, p.27]. According to [2, p.60], software product quality is defined as ``the degree to which a system, component, or process meets specified requirements, customer or user needs or expectations''. Examples of product qualities include safety, performance, and security. Specific techniques are available to achieve product quality in the development of critical systems. In this paper, we call these techniques ``Quality Attribute Techniques''(QAT). Examples of QAT for safety include hazard analysis techniques such as Failure Mode and Effect Analysis (FMEA) and Fault Tree Analysis (FTA).

QAT are used in software development in order to achieve required levels of product quality requirements such as safety and performance. There are many complementary and alternative [3] and it can be difficult to choose the most appropriate techniques. There are some existing guidelines and approaches to help development teams to meet specific product qualities. Examples for safety-critical software systems are [4-8]. Most software processes tailoring methodologies are designed to address issues such as project characteristics, size of the organisations, standard compliance in process tailoring and the technical and social suitability of the organization [9-11] QAT are used but not represented in detail or not incorporated well in software development process models [12, 13]. These guidelines and approaches usually target one specific group of QAT and may not be appropriate to be applied for other QAT. There have been no general approaches proposed for QAT representation and integration with software processes. It is difficult to communicate, monitor and analyse the relationship between QATs and other process elements if QATs are not well presented in process models. A good representation that allows effective ``change'' mechanisms is also needed to support process tailoring, by facilitating the selection of QAT for inclusion into process models that target specific product qualities.

A number of metamodels such as Software Process Engineering Metamodel (SPEM) [14,15] OPEN Process Framework (OPF) [16], OOSPICE metamodel and Standard Metamodel for Software Development Methodologies (SMSDM) [17] have been developed to specify the concepts, rules and relationships used to define software development processes and their components. In this paper, SPEM has been selected to represent the main characteristics of QAT and the relationship of QAT with other process elements. EPF Composer is a process modelling tool based on the SPEM metamodel and is used in this paper to define QAT and incorporate QAT into the OpenUP process model.

The following two research questions are the focus of this study:

- How can QAT be represented using EPF Composer and the SPEM metamodel?
- What are the tailoring features in EPF Composer that can be used to integrate QAT into software process models?

This paper proposes and evaluates two approaches for representing QAT for better integraton of QAT with process models in SPEM and EPF Composer. Safety techniques have been selected as a case example for evaluation. The outline of this report is as follows. Section 2 discusses QAT, SPEM and EPF Composer. Section 3 describes the two alternative representations of QAT. Section 4 discusses the evaluation of the two alternatives. Section 5 discusses the evaluation and suggests some improvements to SPEM and EPF Composer. Section 6 presents the conclusion of this study and discusses future research.

2 Background and Overview

2.1 Quality Attribute Techniques

QAT are techniques that are used to identify, eliminate, reduce, control and minimise potential quality problems in the development of critical systems. For examples, safety critical systems are concerned with a measure of the hazards to life, property or environment, and security-critical systems focuses on resistance to external threats and malicious actions against its integrity [3] and performance-critical systems emphasize response time or throughput [18]. However, QAT are not represented in detail in software development process models. Also, QAT are not well integrated with other process elements such as tasks, roles and work products in different stages of a process model.

QAT for safety-critical system have been selected as a case for initial evaluation in this paper because the area of system safety is well-established. There are many existing procedures, handbooks, standards, books and other references. In safety-critical systems, techniques are available to perform hazard evaluation, control and analysis. A hazard is a state or set of conditions of a system that, together with other conditions in the environment of the system, can lead to an accident or loss event. [3]. Safety techniques can be grouped into different categories such as hazard identification, hazard analysis and safety testing.

An initial investigation has been conducted in safety area to capture characteristics of QAT. Some important characteristics which provide detailed information of QAT and crucial to integrate QAT into process models have been extracted from the literature: [3,19-22]. An example of using these characteristics to describe the FMEA technique is shown below:

- **Technique Name**: Failure Modes and Effects Analysis (FMEA)
- **Aims**: Primarily used for subsystem and system hazard analyses. It also helps to identify critical items in terms of safety and reliability.
- **Descriptio**n: FMEA analyses which failures in a system can lead to undesirable situations. The probability and seriousness of the results for each failure mode are calculated. Corrective actions are prioritised and recommended.
- **Main performer(s)**: Safety Engineer
- **Optional Performer(s)**: System Analyst, Design Engineer, End User, Project Manager
- **Step(s)**: 1. Define the system to be analysed and determine the scope of the analysis. 2. Analyse and organise potential failure modes and their effects. 3. Identify and prioritise corrective action.
- **Input**: Design drawings, functional diagrams, previous analytical data, system descriptions, lessons-learned data, PHL, PHA report
- **Output**: FMEA worksheets, FMEA reports, Critical Item List (CIL)
- **Guidance Documents**: FMEA Worksheet Template, FMEA Guideline
- **As Source Data for (optional)**: Fault Tree Analysis (FTA)
- **Category**: Hazard Analysis
- **Benefits**: Thorough and systematic approach, quickly reveals critical single-point, reliability can be evaluated in detail
- **Limitations**: Does not consider system effects or human error, detailed and expensive to apply to large systems
- **Cost of Application**: Moderate (week)
- **Expertise**: Moderate training (strongly dependent on analyst's understanding of the failure modes)
- **Phase(s)**: Subsystem and system hazard analysis during design phase

- **Team/Individual approach**: Team
- **Single/Multiple Failures Analysis**: Single

Two alternatives are shown in section 3 to represent these important characteristics of QAT and incorporate them into process models.

2.2 SPEM Metamodel

SPEM is a metamodel for defining processes and their components using UML as a concrete notation [14,15]. Fig.1 illustrates parts of SPEM metamodel.

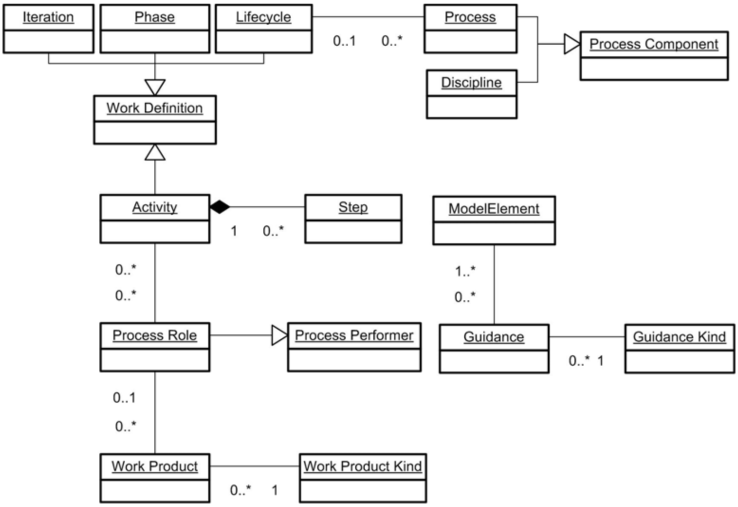


Figure 1: SPEM Metamodel (adapted from [14,17]). ModelElement is a super-class of the other classes in this diagram.

The core idea of SPEM is that a development process is a collaboration of multiple process elements to achieve a specific project goal. A development process *lifecycle* can be structured into *phases* and *iterations*. *Work definition* describes the *activity* (i.e. tasks, operations and actions) performed in a process by a *process role*. An *activity* can be subdivided into *Steps*. *Work products* or *artefacts* (e.g. a piece of code, a document, a mode and source code) are produced, consumed, or modified when mutiple roles interact or collaborate during the process execution. Process components called *Disciplines* are used to categorise activities which share a common theme. Each activity is grouped under a Discipline. A *Guidance* element can be attached to any of the process elements shown in Fig.1 (e.g. *Activity*, *Step* and *Process Role*). More detailed information of a process element can be provided by *Guidance* elements (e.g. checklist, guideline, practice, report and template).

SPEM has been selected to represent the important characteristics of QAT and integrate QAT into process models. Section 3 discusses two alternative representations of QAT based on the SPEM. The ability of SPEM in supporting representation and integration for QAT are evaluated in section 4.

2.3 EPF Composer

EPF Composer is a process modelling tool platform and extensible conceptual framework based on SPEM [15] for authoring, maintaining and customising software development processes [23]. Following SPEM, reusable method content is defined separately from its use in processes in EPF Composer. EPF Composer also provides three sample process frameworks, i.e. OpenUP/Basic, Extreme Programming and Scrum. Users can choose and customise existing process frameworks or create new ones. These frameworks can be transformed into Electronic Process Guides (EPGs). This provides convenient access and assists navigation of the process model for process performers.

In section 3, EPF Composer is used to define two alternative representations based on SPEM for QAT and incorporate QAT into process models. OpenUP/Basic has been selected as an example of process model in this paper. The evaluation of the ability of EPF Composers to support representation and integration of QAT is discussed in section 4.

3 Representation of QAT Using SPEM Metamodel & EPF Composer

3.1 Overview

Two alternatives are shown in this section to represent important characteristics of QAT and integrate them into process models. The first alternative represents a QAT as a *Step* and the second alternative represents a QAT as a *Task*. These two process elements have been selected to represent QAT because the important characteristics of QAT can be represented in more structured formats. Additionally, the relationship between QAT and other process elements can be defined clearly. *Guidance* element and its supported *Guidance Kind* are less structured to represent important characteristics for QAT. As a result, *Guidance* is only used to describe additional information for QAT. The process element *Activity* in SPEM (see Fig.1) is similar to the process element *Task* in EPF Composer. A *Task* consists of a number of *Steps*. Safety techniques are used as case example for these two alternatives.

For both alternatives, a new method plug-in that based on OpenUP is defined to customise method content without directly changing the original content of the OpenUP. It comprises new method content, *Disciplines*, and *Capability Patterns* to represent new safety processes. The new components will be reusable for new releases of OpenUP plug-in. Four kinds of *Content Variability* mechanisms allow process engineers to create new plug-in that *contribute*, *extend*, *replace* or *extend-replace* new definitions to the tasks in the original plug-in [23].

3.2 Alternative 1: Step-Based Approach

In the Step-Based Approach, a QAT is represented as a *Step* in a *Task*. The main characteristics of QAT are added in the the step's description in EPF Composer. Alternatively, information of QAT can be described with *Guidance* documents such as guidelines, templates and worksheets. Hyperlinks to these *Guidance* documents need to be added to description editor for QAT *Step*. If another *Task* also use the same technique, manual duplication of the content is required by adding new *Step* and attaching *Guidance* documents to this Task. *Step* cannot be reused in other *Task*. This approach aimed to incorporate QAT as a step into a task in process models. QAT can also be added as a *Step* in new *Tasks* added to achieve specific product quality goal such as safety and security.

Safety techniques are used as case example for Step-Based Approach. New content packages such as risk assessment, risk control and general safety have been defined to categorise and maintain safety tasks. Under each content package, new safety *Tasks* such as Preliminary Hazard Identification (PHI), Preliminary Hazard Analysis (PHA), Subsystem Hazard Analysis (SSHA) and System Hazard Analysis (SHA) are created to represent safety-critical project specific processes. Safety techniques are added as *Steps* in appropriate safety *Tasks*. Important characteristics captured for each safety technique can be added manually to step's description or provided by using guidance documents such as guidelines, examples and templates.

3.2.1 Tailoring Feature(s) Used

To integrate QAT into process models, the new QAT *Step* is attached to one *Task*. *Content Variability* mechanism ``contribute'' can be used to append a new QAT step to the original content of an existing task without changing original content. The sequence of this step can be changed in the parent task to show the most appropriate order to perform this technique. *Disciplines* have used to categorise *Tasks* that share the common quality goal and use QAT. A new software development lifecycle have been defined to include new QAT *Step*. Three options of process variability are available to copy these processes: Copy, Extend or Deep Copy. ``Deep Copy'' function has been used because the sequence of these tasks can be reorganised after selecting them into *Work Breakdown Structure (WBS)*. As a case example, five ``Disciplines'' are used to group safety project specific processes: hazard analysis, safety coding, safety design, safety review and safety test. A new software development lifecycle, ``SafetyProcessLifecycle1'' has been defined to include existing OpenUP process and also appropriate safety tasks from the five disciplines.

3.2.2 Process view

A new project configuration has been defined to build new process without making changes directly to the original plug-in. A custom category has been added to generate a new process view to customise he process content and methods to be published. For the case example, new safety plug-in and Open-UP plug in was added to this configuration. SafetyProcessLifecycle1', safety disciplines and safety roles have been selected to populate navigation views and new EPG for safety related projects.

3.3 Alternative 2: Task-Based Approach

In the Task-Based Approach, a QAT is represented as a *Task* that describes what needs to be performed by different roles and what work products are used as input and output to achieve a specific quality goal. The main characteristics of QAT is added in different attributes of a *Task* in EPF Composer. Additional information of QAT can be provided by using *Guidance* documents such as worksheets, templates, guidelines and examples. This approach aims to incorporate QAT as a new *Task* in process models. Alternatively, new QAT *Tasks* can extend, contribute or replace original content of existing task in process models.

As a case example, new *Tasks* are created to represent different safety techniques such as FMEA and FTA. New content packages such as risk assessment, risk control and general safety have been defined to categorise and maintain safety technique's *Tasks*.

3.3.1 Tailoring Feature(s) Used

Two ways have been used to integrate QATs *Task* to processes. The first is using *Content Variability* mechanisms to ``contribute'', ``extend'' or ''replace'' QAT information to a base task in process models to avoid modifying the original content. Both content from contributing QAT *Task* and base *Task* will become unified in new browsing view. As a case example, ``contribute'' or ``replace'' mechanisms have been used to respectively append or replace the new content of a QAT task to the original OpenUP content.

Another way of integrating a QAT *Task* to process models is using *Capability Patterns*. These *Process Patterns* can be used to categorise different types of QAT *Tasks*. Additionally, *Capability Patterns* allow process engineers to indicate the flow of tasks and create activity diagrams for project specific processes. They can be reuse and applied to many different lifecycle processes. As a case example, six *Process Patterns* have been defined for safety technique *Tasks*: Preliminary Hazard Analysis, Safety Design, Safety Testing, SSHA and SHA. A new software development lifecycle, ``SafetyProcess-Lifecycle2'' has been defined to include appropriate process patterns, safety technique's *Tasks* and existing process content from OpenUP plug-in.

3.3.2 Process view

In the approach, a new project configuration is defined for new processes without directly changing the original plug-in. A custom category is added to generate a new process view to customise the process content and published methods. In the safety example, a new safety plug-in and Open-UP plug were added to the configuration. SafetyProcessLifecycle2, a new safety process pattern and safety roles were selected to populate navigation views and to create a new EPG for safety-related projects.

4 Evaluation

This section evaluates the capability of SPEM metamodel and EPF Composer in supporting two alternative representations shown in previous section. The evaluation identifies advantages and limitations of the SPEM and EPF Composer.

4.1 Evaluation Criteria

Four criteria have been selected for the evaluation based on the criteria for effective process models defined by Humphrey and Kellner [24]. They highlight that, ``process models must represent the way the work is actually (or is to be) performed, provide a flexible and easily understandable, yet powerful, framework for representing and enhancing the process and be refinable to whatever level of detail is needed'' [24, p.332]. For the purpose of this studies, we focus on how the two alternatives represent characteristics of QAT, how easy to integrate QAT into process models, the reusability of process elements and contents defined by the approach and maintanability to manage the representation and integration.

- Representation: How well the important characteristics of a QAT can be captured with the approach?
- Ease of Integration: Is it easy to integrate QAT to a development process model using the approach?
- Reusability: When using the approach, can the QAT be applied for different processes?
- Maintainability: How convenient is it to manage the representation and integration using the approach?

4.2 Evaluation for Step-Based Approach

- **Representation**: A QAT can be easily added as a simple *Step* in a *Task*. The main characteristics of each QAT can only be added manually in a *Step* description or represented as guidance documents such as guidelines and templates. These guidelines and templates can be attached to the *Task* or put as hyperlinks in the description of the *Step* created to perform this QAT. There is no specific format for the guideline, any information can be entered for a specific QAT. The approach fails to show the relationship of this QAT with other process elements like *Roles* and *Work Products*. Users are not able to have a clear understanding about who will perform this QAT and what kind of input and output artifacts are required for this QAT.
- **Ease of Integration**: Adding *Step* to a *Task* is a simple way to integrate a QAT to a safety *Task* or an existing OpenUP *Task*. If a *Task* needs two different QAT, an additional *Step* to present the second QAT can be added easily to this *Task*. However, duplicate copies of the *Step* need to be created manually for multiple *Tasks* which use a same technique. Additionally, *Content Variability* mechanisms do not allow users to ``contribute'' a new QAT *Step* to more than one existing *Task* in process models. It is also difficult to show the relationships of a QAT with other process elements.

- **Reusability**: New safety *Tasks* can be reused but a *Step* which has been used to integrate a safety technique with a *Task* cannot be shared with other *Tasks*. User must manually duplicate the *Step* and link or attach guidance documents to every *Task* which requires the same technique. Alternatively, user can reuse the same technique by attaching *Guidance* documents of this technique to any Task without adding a *Step*. However, some *Guidance* kinds such as template, roadmap, practice and term definition cannot be added to a *Task* in EPF Composer. Additionally, Content *Variability* mechanisms which are used to ``contribute'' new QAT *Step* to an exisitng *Task* in process models do not allow users to reuse same QAT to multiple *Tasks*.
- **Maintainability**: New safety *Tasks* such as hazard analysis has been defined to represent safety-critical project specific processes when no generic OpenUP *Task* is relevant to the QAT *Step*. Extension or contribution can be done to existing *Tasks* or new safety *Tasks*. It is easier to maintain the project specific *Tasks* by adding extra *Tasks* in new method plug-in.

4.3 Evaluation for Task-Based Approach

- **Representation**: Most of the main characteristics of a QAT can be represented well in a *Task*. Attributes in a *Task* allow process engineers to include detailed information for the QAT such as purpose, main description and steps to conduct this QAT. Additionally, these QAT were able to relate with other process elements such as *Work Products* and *Roles*. A clear relationship can be shown. Process performers are able to find out who will perform this QAT and what kind of input and output artifacts are required for this QAT. However, when a QAT *Task* contributes to an existing *Task*, some of the important information of this QAT like presentation name will not be shown. Process performers may not be able to differentiate the main information of a QAT from the original content of an OpenUP or a safety lifecyle *Task*. Additionally, if more than one QAT contribute to a *Task*; the content of this *Task* will become more complicated.
- **Ease of Integration**: The first way of integration a QAT *Task* to an existing *Task* by contributing, extending or replacing a QAT description to an existing task. However, a QAT *Task* cannot contribute, extend or replace multiple existing tasks. A more flexible method for integration is using *Process Patterns* to include QAT *Tasks* and OpenUP *Tasks*. A QAT *Task* can be applied to different *Process Patterns* which require the same technique.
- **Reusability**: The first way of integration a QAT *Task* to an existing *Task* by contributing, extending or replacing a QAT description to an existing task. However, a QAT *Task* cannot contribute, extend or replace multiple existing tasks. A more flexible method for integration is using *Process Patterns* to include QAT *Tasks* and OpenUP *Tasks*. A QAT *Task* can be applied to different *Process Patterns* which require the same technique.
- **Maintainability**: Another configuration needs to be defined in order to show the *Process Patterns*, QAT *Tasks* and contributed content of a safety technique. This is because the main configuration will only show the original content of the process element. This is a way to avoid overriding the original content of an element.

5 Discussion

Both alternatives have their own advantages and limitations. The Step-Based Approach is a structured and simple way to integrate a QAT to an existing *Task* in a process model. However, adding a *Step* is hard to highlight the main characteristics of a QAT since all the information need to be added as links to guidance documents or entered manually in the *Step*'s description. Additionally, relationships of QAT with other process elements such as work products and roles are harder to recognise since they were added as hyperlinks in the descriptions of the *Step*. The Task-Based Approach which uses a *Task* to show the characteristics of QAT is a more structured way to represent QAT. Relationships of QAT and other process elements such as *Work Products* and *Process Roles* are able to be modelled clearly in EPF Composer. *Process Patterns* are more useful to integrate and categorise QAT in different delivery processes. A *Process Pattern* can be reused in many parts in a delivery process with individual customisation (e.g. remove unnesscary tasks or add new tasks) for the pattern's content.

Based on the evaluation, the main limitation for both alternatives is that some problems exists when manual duplication is required to reuse the same technique for different processes. Process engineers can copy whole or parts of existing processes from a method content library for duplication. However, some of the content (e.g. relationship with other process element) which are inherited from original method contents cannot be modified. Content Variability mechanisms only can contribute, extend or replace content of QAT to a *Task* in EPF Composer. Contribution, extension or replacement of QAT's *Step* or *Task* to multiple *Tasks* in process models are not allowed.

The basic concept of SPEM regarding clear separation of method content from how it is used in processes is useful for the integration of QAT into software development process models. This provides flexibility to incorporate QAT *Tasks* or *Steps* into any part of process models. There are some limitations of existing SPEM metamodel and EPF Composer to support the representation of a QAT and also to incorporate it into process models. Selection strategy and tailoring method for more effective QAT selection and integration with software processes are out of the scope of this paper. The following suggestions may improve this:

- Each Step in a *Task* requires different *Performers*, contribute different *Work Products* to a task and also need different *Guidance* elements to provide more detailed information to practitioners. SPEM can be extended to allow process engineers to assign *Performer(s)*, *Work products* and *Guidance* elements to a specific *Step* in a *Task* in a more structured way.
- A new type of *GuidanceKind* can be added in SPEM for QAT. It can include attributes to represent important characteristics of QAT in a more structured way. This new GuidanceKind should be able to attached to multiple *Tasks* or *Steps* that using the same QAT in processes.
- Multiple tasks or steps may use the same technique. The *Content Variability* mechanisms in SPEM can be extended to support contribution, extension or replacement of a QAT's *Task* or *Step* to multiple existing *Tasks*.
- New attributes can be added to the *Task* element in EPF Composer to represent some important characteristics of QAT.
- EPF Composer can provide a more flexible approach to support the improvements of SPEM, reusability of a QAT and at the same time provide strong capability to maintain the consistency of process models after any change.
- Besides the important characteristics of QAT, *Guidance* documents such as guidelines, templates and examples can be used to provide additional information for practitioners to execute the QAT efficiently and accurately.

6 Conclusion and Future Work

The goal of using QAT is to identify, eliminate, reduce, control and minimise potential quality problems in the development of critical systems. QAT are used to improve product qualities such as safety, performance and security. However, QAT are not well integrated into software development process models. A good representation allows development teams to communicate, monitor, control and analyse the integration of QATs with software processes effectively. This study has investigated how the SPEM metamodel and EPF Composer can support the representation of characteristics of QAT for better integration into software process models such as OpenUP. Two alternatives - a Step-Based Approach and a Task-Based Approach have been proposed for the representation of QAT and integration with software process models. Safety techniques were selected as a case example for an evaluation. Based on the evaluation, there are several benefits and limitations of the SPEM metamodel and EPF Composer in supporting the two alternatives. The concept of separation of method content from processes in SPEM is useful to define charaterisctics of QAT and incorporate QAT into different processes in process models. However, there are still some limitations especially when the process content of QAT are manually duplicated to be reused for different processes. Some content inherited from the original method content cannot be modified.

Process models which are defined based on metamodels are expected to be more extensible, reusable, configurable and tailorable to specific project needs. Some of the future challenges of metamodels and process modelling tools are to include providing improved support for process component reuse, representation of QAT and integration of a process element into process models. Future work of this research are developing a tailoring method to incorporate appropriate QAT into software development process models. SPEM metamodel and EPF Composer will be extended to support the integration approach.

7 Literature

[1] Forrester, E.: A Process Research Framework. The International Process Research Consortium (IPRC) (2006)

[2] IEEE Std 610.12-1990: IEEE Standard Glossary of Software Engineering Terminology. Institute of Electrical and Electronics Engineers (1990)

[3] Leveson, N.: Safeware: System Safety and Computers. Addison Wesley (1995)

[4] Lutz, R.R.: Targeting Safety-Related Errors During Software Requirements Analysis. In: ACM SIGSOFT Software Engineering Notes. (1993) 99-106

[5] Lawrence, J. D.: Software Safety Hazard Analysis Version 2.0. Technical report, Lawrence Livermore National Laboratory (1995)

[6] Joint Software System Safety Committee: Software System Safety Handbook. A Technical & Managerial Team Approach. (December 1999)

[7] Bhansali, P.V.: Software Safety: Current Status and Future Direction. SIGSOFT Software Engineering Notes 30(1) (2005) 3

[8] Borcsok, J., Schaefer, S.: Software Development for Safety-related Systems. International Conference on Systems (ICONS'07) (2007) 37-42

[9] Basili, V.R., Rombach, H.D.: Tailoring the Software Process to Project Goals and Environments. In: International Conference on Software Engineering (ICSP). (1987) 345-357

[10] Bowers, J., May, J., Melander, E., Baarman, M., Ayoob, A.: Tailoring XP for Large System Mission Critical Software Development. In: XP/Agile Universe Conference on Extreme Programming and Agile Methods, Springer-Verlag (2002) 100-111

[11] Pedreira, O., Piattini, M., Luaces, M.R., Brisaboa, N.R.: A Systematic Review of Software Process Tailoring. SIGSOFT Softw. Eng. Notes 32(3) (2007) 1-6

[12] Zhu, L., Jeffery, D.R., Staples, M., Huo, M., Tran, T.T.: Effects of architecture and technical development process on micro-process. In: International Conference of Software Process (ICSP). (2007) 49-60

[13] Zhu, L., Tran, T.T., Staples, M., Jeffery, D.R.: Technical development process in the xml domain. In: International Conference of Software Process (ICSP). (2009)

[14] OMG: Software Process Engineering Metamodel (SPEM) Version 1.1 (2005)

[15] OMG: Software Process Engineering Metamodel (SPEM) Version 2.0 (2008)

[16] Firesmith, D., Henderson-Sellers, B.: Safeware: System Safety and Computers. Addison Wesley (1995)

[17] Henderson-Sellers, B., Gonzalez-Perez, C.: A Comparison of Four Process Metamodels and the Creation of A New Generic Standard. Information & Software Technology 47(1) (2005) 49-65

[18] Smith, C., Williams, L.: Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software. Addison Wesley (2002)

[19] Storey, N.: Safety Critical Computer Systems. Addison Wesley (1996)

[20] Stephans, R.A.: System Safety for the 21st Century. Wiley (2004)

[21] Vincoli, J.W.: Basic Guide to System Safety. Wiley (2006)

[22] Zurich Risk Engineering: Which Hazard Analysis? - A Selection Guide (1998)

[23] Haumer, P.: Eclipse Process Framework Composer: Part 1 and Part 2. Technical report, IBM Rational Software (2006)

[24] Humphrey, W.S., Kellner, M.I.: Software Process Modeling: Principles Of Entity Process Models. International Conference on Software Engineering (1989) 331-342

8 Author CVs

Yin Kia Chiam

Yin Kia, Chiam is a PHD student of the Computer Science and Engineering Department at the University of New South Wales. She is also a lecturer of the Computer Science department at the University of Malaya. Before pursuing her PHD, Yin Kia had position as a Software Test Engineer in Motorola Malaysia Software Center. She holds a MSc in Information Technology from Malaysia University of Science and Technology (MUST), which is a collaborative programme between MUST Ehsan Foundation and the Massachusetts Institute of Technology (MIT).

Mark Staples

Dr. Mark Staples is a Research Group Manager at NICTA, Australia's ICT Centre of Excellence. Before joining NICTA, Mark had positions in the software engineering industry as a Process and Tools Manager at a medium-sized enterprise software development company, and as a Configuration Manager and Test Engineer in the development of the safety-critical environmental control system of a mass transit system. He is a Conjoint Senior Lecturer at the University of New South Wales, and holds a PhD from the University of Cambridge.

Liming Zhu

Dr. Liming Zhu is a researcher in the Software Engineering group at National ICT Australia (NICTA) and teaches software engineering at University of New South Wales. He formerly worked in several technology lead positions in software industry before pursuing a PHD degree in software engineering at University of New South Wales (UNSW). He has worked with Australia government agencies, defense, standardization bodies and commercial companies. He is a committee member of the Standards Australia IT -015 (system and software engineering) group. He contributes to the ISO/SC7 Working Group 42 on architecture related ISO standards. His research interests include software architecture, service engineering, model driven development, business process modeling and software development processes.

Acknowledgements

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

Addressing IEC 61508 Certification in a multi-standards context: A generic approach

Gautier Dallons[1], Annick Majchrowski[1], Jean-Christophe Trigaux[2]
[1]CETIC, Charleroi, Belgium
[2]FUNDP, Namur, Belgium
{gautier.dallons, annick.majchrowski}@cetic.be
jtr@info.fundp.ac.be

Abstract

Software embedded in safety equipment often requires certification for its deployment in critical systems such as trains, airplanes, etc. The selection of a certification standard for the software depends on the marketplace of the respective critical system. For example IEC 61508 is a generic safety standard used in a large variety of markets and DO-178b is specific to aeronautic market. Multi-standards certification is therefore a key challenge for organization producing this kind of software. Currently, there is no rational method to reduce the costs incurred in the multi-standards certification. This paper addresses this problem by presenting a generic approach for reducing certification costs in the context of multi-standards certification.

Keywords

Certification, IEC 61508, DO-178B, standards, Gap analysis

1 Introduction

Enterprises developing software for safety equipments require certification of their products in order to assure the functioning of their safety features. A market-specific certification of these products implies the choice of an appropriate standard. However, the selection of such a market-specific standard puts a ceiling on company's access to other markets that require different certification… For example, the IEC 61508 safety standard [1] is not defined in a way that makes this standard compatible with other safety standards such as DO-178b standard [3]. This situation gives rise to a new challenge for companies to adopt a certification process that compliant their products to multi-standards.

The problem of multi-standards certification is not fully addressed in the existing literature. The link between the certification of a critical system and the software embedded in this system is presented in [5]. However, it does not address the problem of certifying the software following different standards. The reusability in the context of a specific standard is addressed in some papers [6, 7, 8], nevertheless extending the reusability to multi-standards has not been addressed. Cost reduction of certification is explored in [9] that present a formal gap analysis to identify alternate strategies for reducing the cost of compliance). But the certification in a multi-standards context is still a barren land in the need of cultivation.

We have designed a conceptual framework to find a way out of this impasse. Our proposed framework compares and brings closer various safety standards. This framework is used in IEC 61508 [1] and DO-178b safety standards [3]. Based on this comparison work, we have built a certification methodology for a product following IEC 61508 requirements and is compatible with the future DO-178b certification. This methodology is supported by a gap analysis and a tracing tool.

The contribution of this paper is to propose a methodology to address multi-standards certification. The rest of the paper is organized as follows. Section 2 describes the comparison method used in this work. Section 3 describes the conceptual framework defined to bring closer the standards. This framework is used as a mean to compare different standards. Based on this framework, we propose an alignment between IEC 61508 and DO-178B safety standards in the section 4. This alignment allows us to define common requirements and variations between these standards. Section 5 presents the methodology to address multi-standards certification. Some conclusions are drawn together with a description of our future directions in the section 6.

2 Comparison Method

This section presents the method we have used to compare different safety standards according to a generic quality assessment framework that was particularly instantiated for safety critical software. First, the general approach behind this framework is described. Secondly, we explain how it can be used in practice for comparing different standards.

Our aim is to instantiate a generic quality assessment framework for safety critical software. This framework should have to fix the terminology, to compare different standards and to identify the conflicts between what is advocated by the standards and the ground situation in a company. Moreover, the framework should continuously evolve with the standards and the new business needs.

The framework (Figure 1) is decomposed into three levels, each corresponds to a specific concepts: Quality assessment-related concepts, Software-related concepts and Safety-related concepts. We do not assume that the defined concepts are complete or without ambiguities. We claim that comparing those concepts with concepts proposed in different standards will help to standardize the terminology and consequently to make the framework evolve. In addition, the framework concepts are not defined from scratch. They are based on existing terminologies. For instance, the terminology defined in the IEC12207 International Standard [2] is extensively reused to define software-related concepts.

In practice, this framework is used to analyze and compare different software quality standards with their respective terminologies and principles. On the other side, it also allows relating company specific concepts to concepts defined in different standards. It allows aligning concepts, studying their differences and if necessary updating the concepts' definitions proposed in the framework. In addition, it facilitates the understanding of the standards and helps to position them with respect to each other.

2.1 Map one standard to the framework

The definition of framework concepts is followed by its instantiation with the standards. These standards can be international or local ones. The framework instantiation is realized according to the context where quality should be assessed. The quality assessment process or certification process is strongly dependant on the context. Many factors may influence this process such as the company; the projects; the products; the organization; the people; and their responsibilities.

It is quite difficult to identify which concept in the standard can be an instance of a concept in the framework. We call it concept alignment. This alignment of concepts is complex and quite subjective. One method to facilitate this task is to analyze the definitions of the respective concepts defined in the different documents and to position them in accordance with the framework structures and the standard. However some concepts can not be aligned to the framework. The main reasons are either that the incriminate concept does not exist in the framework or that the incriminate concept partially covers one or several concepts of the framework.

2.2 Comparison of different standards according to the framework

Once the framework has been instantiated with several safety standards, their respective concepts can be aligned using the concepts of the framework as a backbone for the alignment. This allows the identification of gaps either in the standards or in the framework itself. In addition overlapping between standards can be identified. In practice, the main job consists of aligning quality requirements that the different standards suggest to follow. It helps to identify those requirements that are common between standards and those requirements that are variable. According to the structure of the framework it also helps to identify the quality aspect a standard is more constraining than other ones and vice versa.

3 Generic Software Safety Quality Assessment Framework

This section describes our proposed framework and highlights its main concepts. We use a top down approach to present the framework. General concepts related to the quality assessment are presented in the section 3.1. Their refinement is shown in the section 3.2 with software-specific concepts. Finally, both general and software specific concepts are refined into safety concepts in the section 3.3.

3.1 General concepts

The main purpose of a quality assessment (or certification) is to verify that the quality target (in particular a certification target) is compliant with the goals specified in a standard in a specific quality context (in particular a certification context). Standards often refine these general goals into more specific and precise requirements. Requirements may concern one artefact or may be transversal to many. Artefacts are essentially decomposed into two categories: Product and Process. A product is the result of an activity. A process is defined as a succession of activities. Evidences should be provided to assess that the goals are fulfilled and thus the corresponding requirements are satisfied. The evidences assess that the right techniques have been effectively and correctly used to satisfy the requirements. These evidences are then gathered within a certification report. These concepts are represented in figure 1 by simple-lined rectangles.

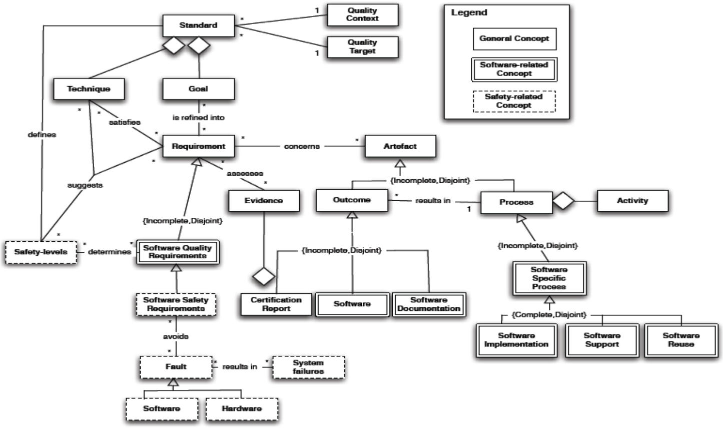


Figure 1 : Safety conceptual framework

3.2 Software-related concepts

The aforementioned concepts are general enough to cover quality assessment of any process or product. Our aim is to reduce the scope of this quality assessment framework so that software safety can be effectively addressed.

Software is the program and other operating information used by a computer. It is an outcome of the development process. Software information is grouped in several software documentation files. Software is implementing requirements and respect quality requirements. Software specific processes can be classified as software implementation (Requirement analysis, Architectural Design, Coding, etc.), software support (Documentation management, Quality assurance, Configuration management, etc.) and software reuse (Domain engineering, Reuse program management, Reuse asset management). These concepts are illustrated in figure 1 by double-lined rectangles.

3.3 Software-related concepts

Safety is defined as “a property of a system that it will not endanger human life or the environment” [4]. Whereas certification concerns with safety-critical systems’ other issues that should be addressed with specific concepts.

The first issue concerns an activity that should be preliminary to safety critical software development. This activity is named as safety analysis (i.e. hazard analysis) and consists of (1) determining the risks associated to the actual or potential danger to people or the environment, (2) selecting an appropriate safety level and (3) specifying appropriate development methods (techniques) to manage those risks.

The second issue concerns the concept of fault. A fault is a defect within the system. Software faults can be decomposed into specification and coding faults. Essentially, it includes incorrect or incomplete specification, stack overflows, use of uninitialized variables... Faults in a system are often inevitable and considered according to three different perspectives.

1. For highly critical systems, no faults are tolerated and expensive techniques are employed to guarantee this situation;
2. For less critical systems, faults may appear, however, system failures should be avoided;
3. For other systems, faults may result in system failures; however, the associated risk should be acceptable for the safety-level.

Once the faults have been identified, quality requirements should be defined to avoid, detect, remove or tolerate them. In general, standards define a set of quality requirements that should be satisfied in order to (1) drastically decrease the number of occurrences of faults and to (2) obtain the corresponding safety level. Quality requirements may concern various types of software artefacts. Classical examples of safety requirements are: reliability, availability, failsafe operation, system integrity, data integrity, system recovery, maintainability, dependability. In addition, other specific safety requirements should be devised according to the characteristics of the system. The systems should be studied carefully during safety analysis and all the dangers that could result from its use should be identified. Safety requirements are then defined to avoid dangerous situations and/or how to control them. These concepts are presented in figure 1 by dashed rectangles.

4 Standards alignment

Based on the conceptual framework presented in section 3, we have analyzed and compared two different international certification standards applicable to safety-critical software development: IEC 61508 [1] and DO-178B [3]. In this section, we present these standards and the mapping on some important concepts as outcome such as software quality requirements and so on.

4.1 IEC 61508 description

The international standard IEC 61508 [1] Functional safety of electrical / electronic / programmable electronic safety-related systems (E/E/PES) is intended to be a basic functional safety standard applicable to all kinds of industry. IEC 61508 defines functional safety as: part of the overall safety relating to the EUC (Equipment Under Control) and the EUC control system which depends on the correct functioning of the E/E/PE safety-related systems, other technology safety-related systems and external risk reduction facilities. Safety and risk mitigation should be considered during the early phases of development projects. The risk is a function of frequency (or likelihood) of the hazardous event and the event consequence severity. A zero risk can never be reached. However, risk should be reduced to a tolerable level by applying safety functions or adopting safety principles. IEC 61508-3 is specifically dedicated to the definition of safety requirements for the development of software. This part of the standard proposes different techniques to reduce the risk according to risk level. Theses levels are called the Safety Integrity Level (SIL).

4.2 DO-178b description

The DO-178B [3] guidelines essentially describe (1) which airworthiness requirements should be satisfied and (2) which certification documents should be produced to obtain the DO-178B certification. The main requested documents are: the descriptions of objectives for software life cycle processes, the descriptions of activities and design considerations for achieving those objectives and the descriptions of the evidence which indicates that the objectives are met.

Different software levels are defined that allow determining degrees of necessary rigor during the software development process. Annex A of DO-178B/ED-12B [3] enlists the objectives that must be met for each specific software level. The difference in rigors is determined by the number of objectives that need to be satisfied, whether a specific objective is satisfied with independence, and the formality of configuration management of the software data produced during development. The number of objectives to satisfy increases for each safety level, starting from 0 objective (Level E) up to 66 objectives (Level A) to satisfy.

4.3 Standards mapping

The framework is based on generic concepts instanced in the standards. We have considered software and safety concepts of the framework in order to define the mapping. So, safety levels, processes outcomes, software specific processes and software quality requirements have been suited in each standard and are mapped.

4.3.1 Level alignment

DO178b standard define five software levels based on the consequence for the aeroplane and the people onboard. These five levels are:

- level A for catastrophic consequences: Failure conditions that would prevent continued safe flight and landing.
- level B for severe consequences: Failure conditions which would reduce the capability of the aircraft or the ability of the crew to cope with adverse operating conditions to the extent that there would be: higher workload, serious or even potentially fatal injuries to a small number of people onboard, etc.
- level C for major consequences: Failure conditions which would reduce the capability of the aircraft or the ability of the crew to cope with adverse operating conditions to the extent that there would be, for example, a significant increase in crew workload or discomfort to passengers,

possibly including injuries.

- level D for minor consequences: Failure conditions which would not significantly reduce aircraft safety, and which would involve crew actions that lie well within their capabilities.

- level E for no consequences: Failure conditions which do not affect the operational capability of the aircraft or increase crew workload.

IEC 61508 standard defines four safety integrity levels (SIL1 to SIL4). For each consequence considered in a particular domain, a safety integrity level is assigned for each couple of probability of unwanted occurrence, frequency, probability and avoiding the hazardous event. An example of SIL determination is given in the table 1. For example, if we consider a set (C1,F2,E1,P1) the integrity level associated is SIL1.

Table 1. SIL determination example

Consequence	**Frequency**	**Avoiding**	**Probability of unwanted event**	
			P1	**P2**
C1	F1	E1	n/a	SIL1
	F2	E1	SIL1	SIL2
		E2	SIL3	SIL4

Base on table 1, the mapping is trivial. For each consequence considered in SIL determination, we have to determine to which DO-178B consequence category the consequence belongs. For example, in table 1, if consequence C1 could cause a plane crash, it could be mapped to catastrophic category. Then, the determination of DO-178B level is easy considering the DO-178B definition of level.

4.3.2 Outcome alignment

To align the two standards, we have considered each definition of the outcomes specified in the standards. Table 2 gives a subset of the mapping. Following three situations have taken place:

1. The definitions were closely similar. In this case, the mapping is trivial.

2. Only one of the standards has specified an outcome. In this case, the mapping is incomplete due to a lack of definition in one of the standards.

3. The definitions are quite similar but some differences exist (italic concept in the table). In this case, the mapping considers only the common element. The mapping becomes more generic and some particular details are dropped.

Table 2: Outcome mapping subset

DO 178B	**IEC 61508**
SW lifecycle	*SW safety lifecycle*
Airworthiness requirements	Safety requirements specification
SW Requirements Data	SW safety requirements specification
Plan for SW Aspects of Certification	SW safety validation plan
Design Description	SW architecture design description
SW Verification Cases and Procedures	SW system integration test specification
/	Development tools
SW Requirements Standards	/

4.3.3 Processes alignment

The alignment of the processes is quite similar to the alignment of outcomes. The three basic situations occurred also in the case of process mapping. Table 3 gives an example of the mapping. A new situation took place in this study: process output/input differences. We decided not to treat this case because the mapping of outcomes take into account information split in different documents and the main focus of this comparison is the objectives of the process.

Table 3. Processes mapping example

DO 178B	IEC 61508
Requirements Engineering	*SW safety requirements specification*
SW design	SW design and development; SW architecture
/	SW Safety Validation
Certification liaison	/

4.3.4 Requirements alignment

Requirements have been compared in order to align them and to propose a generalization. Table 4 shows an example of requirement alignment. Each standard has a specific interpretation of requirements. To tackle this problem we have chosen to show to the evaluator the generalization form and the specifics forms of the requirements in order to correctly translate the situation analyzed.

Table 4. Requirements alignment example

Generalization	DO 178B	IEC 61508
Software requirements are analyzed for correctness and testability	• High-level requirements are consistent. • High-level requirements are verifiable. • System functional and interface requirements that are allocated to software are analysed for ambiguities and inconsistencies.	• SW safety requirements specification shall be clear, precise, unequivocal, verifiable, testable, maintainable and feasible, commensurate with the safety integrity level

5 Generic process and tool support

This section shows how the comparative analysis of the standards can be beneficial for companies that need certification. Most organizations face strong difficulties to obtain a certification, to understand the requirements of a standard, to align the company's functioning with the standard and to give strong assurance that the company conforms to the standard. Once the basic concepts of the standards are understood, they can be implemented within the company.

Based on the results of the alignment of the two standards IEC 61508 and DO-178B presented in the section 4, we have built a generic process to reduce the costs incurred by a certification in the multi-standards context:

- Cost or overheads generated by the bad initial choice of the target standard. The efforts required for the conformity of the product are perhaps higher than their anticipated return. To have an estimation of the current level of conformity compared to the considered standards allows to make the adequate decisions in order to reduce this risk

- Cost of setting in conformity. To reuse a maximum of existing elements makes it possible to reduce this cost. However, it is necessary to identify these existing elements for their reutilisation later.

- Cost of setting in conformity for another standard.

Our proposed generic process is composed of three consecutive steps: the gap analysis between existing elements and various standards of reliability elements, restructuring of the existing elements to be conformed with the selected standard and the setting in conformity to the selected standard. To facilitate the application of this generic process, we propose a tool for gap analysis.

5.1 Gap analysis and tool support

The first step called “gap analysis” consists of evaluating the difference between existing elements and expected elements according to various reliability standards. It provides an idea of the variations between the standards’ recommendations and the company’s current processes and thus, to have an idea of investment which will be necessary to be conformed to various standards. Before starting a standard specific gap analysis we propose to first position the company according to the different standards in a relatively higher point of view.

To achieve this goal, we ask the company to fill a questionnaire built upon the framework and the standard alignments. First specific question are derived from standards. Then considering the mapping through the framework, specific questions of different standard are generalized. Series of questions are proposed for each software process described in the framework. Table 5 gives an example of the questions. The evaluation of answer is performed by an evaluator considering a textual summary of the answer and using a scale of evaluation.

Table 5 : Example of questions

Generalization	DO 178B	IEC 61508
Do you specify and document the requirements of your software? Are your requirements accurate and complete?	Are low-level requirements consistent? Are high-level requirements consistent?	Are the specified requirements for software safety clear, precise, unequivocal, verifiable, testable, maintainable and feasible, commensurate with the safety integrity level?

Once this first high level gap analysis has been performed, we can identify the SW processes that require drastic changes. Moreover, if we want to go into more details, to focus on one specific standard, the task is facilitated since we keep the traceability links between the detailed requirements of each standard and the general questions. In practice, the questionnaire is defined in an excel sheet. One specific page is dedicated to the questionnaire and specific pages are dedicated to the different standards. In our case the standards are DO-178B and IEC61508-3. Once the general questionnaire is filled, the satisfaction values of each corresponding quality requirements are automatically updated in the standard specific pages. However, some satisfaction values need to be manually filled as they are completely specific to one standard. In addition, the satisfaction values can still be manually refined to produce a finer gap analysis. Finally the results are presented within synthetic graphics (Figure 2).

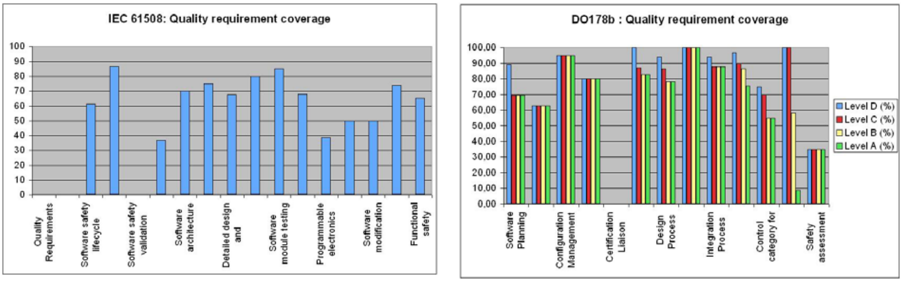


Figure 2 : Gap Analysis

5.2 Restructuring

When a target standard is selected, the second step consists of restructuring the reusable elements which were identified in the first step in order to fulfil the requirements of the selected standard. This reorganization can be facilitated by the traceability information available in the gap analysis. Templates of documents were also created in order to present available information according to the specific vision to each standard. These templates were completed by information directed towards the drafting of contents according to the requirements of the referring standard. These drafting instructions are useful for the third step.

5.3 Setting in conformity

The third step consists of setting in conformity according to a selected level of certification. With this intention, the drafting instructions make it possible to check the constraints to be respected according to this selected level. In the same way, these instructions give the directives of drafting for missing information.

Thus, at the end of these three steps, the elements related to the product to be certified could be put in conformity compared to a particular standard by avoiding their rewriting. This operation can be repeated to set itself in conformity with a second standard.

6 Conclusions and perspectives

The generic certification process is promising in a context of multi-standards. The amount of work needed to make a product compliant is insignificant due to the reusability of existing materials. During the gap analysis phase, the traceability mapping between standards' requirements and existing materials is created. This allows reformatting the documents followed by the requirements of the target standard. This is still a manual step but helpful for traceability information.

Our future directions include creation of a semi-automatic tool for making the transformation based on our conceptual framework. The transformation of document could be made simpler by a "press button approach" exploiting the mapping between document and the generic model.

A second extension of our work concerns the supported standards. Generic nature of our framework enables the tool's extensibility to support new safety standards. The whole IEC 61508 family (IEC 61511, IEC 62061, IEC 61513, etc.) could also be supported. This extension will be useful to broaden the scope and applicability of our proposed methodology.

Our current work does not consider the hardware part of standards. For example, IEC 61508 has a number of requirements about hardware. We have only considered the software part of the standards. In the same way, a dual standard of DO-178b exits about hardware requirements: D0-254. Extending our framework to the hardware part is interesting in order to reduce the total efforts required for the certification of embedded systems. We plan to investigate it in the future.

As a conclusion, our work has shown its pertinence for industry. It helps companies to reduce the amount of work needed to certify a product. Indeed, using this approach, we have mapped existing material that meets particular standard requirements to a target standard. This work has increased the reusability of project documentation and has given a systematic way to address translation to a target standard. Extensions of this work could significantly improve the approach and could further reduce the certification costs by automation of several reformatting tasks.

Literature

[1] I. E. Commission. IEC 61508-3: International standard functional safety of electrical/ electronic/programmable electronic safety-related systems – part 3: Software requirements (1998)

[2] INTERNATIONAL STANDARD ISO/IEC12207 IEEE Std 12207-2008 Second edition.

[3] N. W. S. . W. D. . R. I. . C. A. N. W. S. . W. D. . RTCA, Inc.: DO-178b: Software considerations in airborne systems and equipment certification (1992).

[4] N. R. Storey. Safety Critical Computer Systems. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1996.)

[5] Rodriguez-Dapena, P., Software safety certification: a multidomain problem, IEEE Volume 16, Issue 4 (1999)

[6] Hoyt Lougee, Reuse and DO-178B Certified Software, Foliage White Paper

[7] Hoyt Lougee, DO-178B Certified Software – A Formal Reuse Analysis Approach, Foliage White Paper

[8] Thomas, W. M. and Cerino, D. A. Predicting software quality for reuse certification. In Proceedings of the Conference on Tri-Ada (1995)

[9] Hoyt Lougee, The Road to RTCA/DO-178B, Foliage White Paper

Acknowledgements

This research project Skywin-Telecom is supported by the Walloon Region under the terms defined in the Convention n°5545.

Author CVs

Gautier Dallons

Gautier Dallons is a project manager and team coordinator at the Centre of Excellence in Information and Communication technologies (CETIC) in Belgium. He is expert in software safety and security. He works on improving safety and security through strong process and software certification. He is involved in several research projects and local industrial services.

Annick Majchrowski

Annick Majchrowski is currently working in the role of researcher/senior project manager at the Centre of Excellence in Information and Communication Technologies in Belgium. As researcher, Annick works on the domain of software and system engineering where she focuses on the quality of software products and software processes. Her research interests are in software process assessments and improvements. As senior project manager, Annick is responsible for service missions in companies with focus on quality management in IT domain.

Jean-Christophe Trigaux

Dr Jean-Christophe Trigaux is researcher at the University of Namur (CETIC) in Belgium. He is a member of the PReCISE research group. He is expert in software reuse. He works on improving reusability through product lines and modelling product variability. Recently, he applies is field of expertise in the case of safety critical system certification.

Improving the Software-Development for multiple Projects by applying a Platform Strategy for Mechatronic Systems

Volker Bachmann, Richard Messnarz

Abstract

Approximately 25 years ago the first electronic control units were introduced in the market and have since then built a new field in engineering that was called mechatronics for the first time in the late 1980s. Since then the complexity has increased continuously doubling the lines of code in the control units roughly every 5 years. In order to handle this complexity there were developed new processes for the software development to be able to keep the quality leading to a new European Standard ISO 15504 (SPICE) [1]. About 5 years ago, after it became more and more obvious that looking at software-processes alone is not enough any more, the standard had to be increased to the whole system (or product). This involved the complete mechanic and electronic development. The amount of documentation has come to a point that is not acceptable for development departments anymore. Therefore it is time to think about not only re-use in product development and testing results but also in documentation and process development results.

This paper describes a solution that is based on the principle of a platform strategy that is quite similar to what has been developed originally on a product level in the automotive industry some 15 to 20 years ago. It shows solutions for products for a number of different customers for documentation and development processes on a system level as well as re-use of software products (programs) for different customers and different products.

1 Introduction

The automotive industry is getting under an ever higher pressure to use the given resources in a highly efficient way. For the development of new products this means especially a reduction of development time and time to market. At the same time the complexity of products is rising because of the introduction of electronics and software to raise comfort and reduce energy consumption. Looking at the past 20 years the lines of code in a transmission control unit for an automatic full transmission grew with the factor 10 every 5 years. Therefore the lines of code coming from 1000 some 20 years ago are approaching 10 million. At the same time the number of control units in a vehicle has changed from basically 2 or 3 (engine, transmission, brakes) to more than 100.

In order to master the strongly rising complexity, advanced methods for the project management and product development were developed. One of these methods has become an European standard ISO 15504 (Spice). This standard – originally developed for software development – helps to handle the complexity of the interaction between software, electronic hardware, and mechanics / hydraulics / pneumatics. Every developer who has been confronted with these methods and still remembers the old days of either pure mechanics or just little software involved, will agree, that the amount of documentation reached a significant part of his time.

On the other hand the testing has reached a dimension that requires not only a test department working with test rigs and vehicles but another test department working with software tests and software integration tests doubling the number of testers needed, and still often leaving a feeling that too little tests have been carried out by the time of delivery to the customer.

Since software controlled systems can not be tested by putting them for a given time on a test rig, the European OEMs can not prove the modern systems by looking at this kind of test results. On the other hand the over all number of tests carried out for software controlled systems makes it also impossible for the OEM to check whether or not his supplier has done his job sufficiently. The same problem arises all the way down the supply chain for every one who orders any kind of system (e.g. a supplier who builds transmissions and buys the control unit with an operating system on it).

The OEMs therefore started to carry out assessments with their suppliers to find out to which degree (level) they work according to the standard [2]. Although during this assessment not all the code can be reviewed or every calculation can be checked, they assume that working according to the standard at a level of minimum 2 will lead to a mature product. This way the OEMs force their suppliers to develop and document every product to the required degree.

2 Business environment

In most cases the development of a complex product must be based upon a customer order since the development costs are too high compared to the risk of not developing for the right market. The development then starts for and together with a so called “lead customer”. The lead customer usually hands over a technical specification which is more or less precise. In an automotive environment the specification for diagnosis and CAN-communication is e.g. often down to the most detailed description level, whereas an overall system description is often very general and left to the supplier for precise wording in large areas.

Based upon the customer description a tree of requirements is broken down linking all customer requirements to a system description (so called system requirements) which are linked to subsystem requirements which again are linked to software- / electronic hardware- / and so on –requirements. The vocabulary used for the description is usually taken from the lead customer since it makes communication with the lead customer easier.

After the product has gone through a successful start of production, the wish – if it was not there from the beginning – comes up very quickly, to sell this product to more than one customer. The effort needed for this can be called an adaptation development and in order to earn money should be far less extensive than the original product development.

The key phrase used in this context is “platform strategy”. A platform strategy promises to develop a part of a product only once and re-use it over and over again in multiple products for multiple customers. But is this possible for such different things as software tests, documentation or cogwheels?

3 SPICE and the HIS Scope

According to the „Hersteller Initiative Software“ HIS (OEM initiative software) [3] a reduction of the original number of processes of the standard was determined. The tailoring includes the following processes:

ACQ.4 Supplier monitoring

ENG.2 System requirements analysis

ENG.3 System architectural design

ENG.4 Software requirement analysis

ENG.5 Software design

ENG.6 Software construction

ENG.7 Software integration

ENG.8 Software testing

ENG.9 System integration

ENG.10 System testing

MAN.3 Project management

SUP.1 Quality assurance

SUP.8 Configuration management

SUP.9 Problem resolution management

SUP.10 Change request management

These processes are mandatory in most automotive applications. Each one of the processes has influence on one or more departments of a company. An analysis of the impact on different departments in different phases of a product lifecycle is useful to understand at which point of time who has to do what to minimize the work load of the department. This is because each department relies on the outcome of processes and is not always the one who is responsible to produce these outcomes.

In case the development is performed according to the V-Model some things are very obvious:

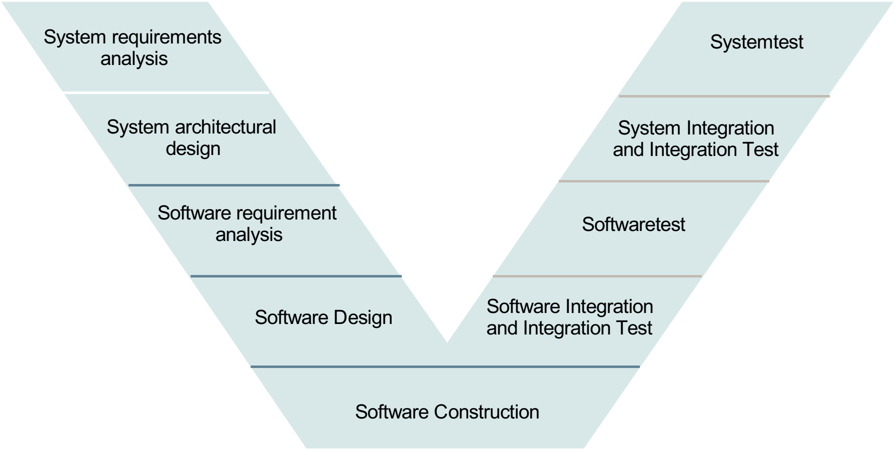

System requirements analysis
System architectural design
Software requirement analysis
Software Design
Software Construction
Software Integration and Integration Test
Softwaretest
System Integration and Integration Test
Systemtest

Figure [1]   V-Model

E.g. ENG. 2 System Design is carried out by the system developer. The result is an input for ENG. 3 Software Requirements Analysis which is usually worked out by a function developer. This can only be done after the system decision in which field (software, electronic hardware, mechanics) the system design requirement should be solved.

This – of course – is a very simple and obvious example. But there are some which are less obvious yet equally important.

Besides the v-model the following processes influence with their outcome these departments:

SUP.1 Quality assurance	quality-management
SUP.8 Config. management	project management
SUP.9 Problem resolution management	project management
SUP.10 Change request management	engineering, controlling
MAN.3 Project management	management,
ACQ.4 Supplier monitoring	controlling, engineering, management

In order to be able to work according to the v-model it is useful to work with the help of a tool. The tools on the market are usually a database with frontend. This makes clear, that a tool that is a table calculation program is not sufficient. The bilateral traceability required by the standard inhibits that.

In these database programs the product requirements are usually represented by a tree structure that follows the decision levels.

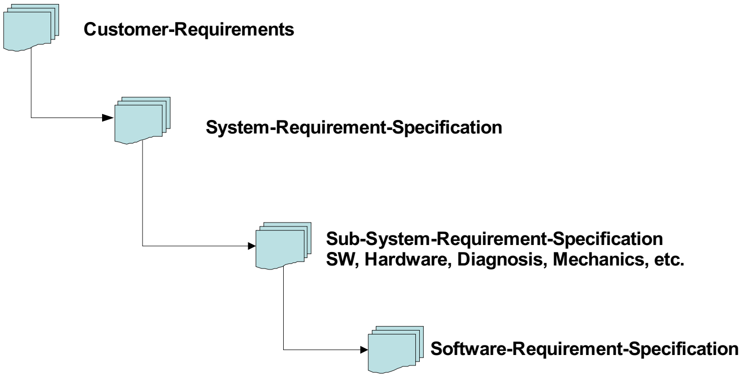

Customer-Requirements
System-Requirement-Specification
Sub-System-Requirement-Specification
SW, Hardware, Diagnosis, Mechanics, etc.
Software-Requirement-Specification

Figure [2] different levels of requirements

Every time the customer comes to assess a project, he wants to see this kind of tree to show, how his requirements are broken down to different fields of development. Stepping from one level to the next always needs a decision on how to realize the requirement and also where. Whereas "where" expresses simply: Is a certain requirement being realized by developing a Software-function or is something being designed, etc.? Therefore it makes sense to split up the sub-system into as many assembly groups or functional groups as the product requires [4].

Since each customer wants to see his requirements on the top-level there is usually one tree found per product and per customer.

3.1 One Product for multiple Customers

The reason for this is because every so called adaptation development is under as high a time pressure as the original development project, and it is always easiest to adapt to the customers vocabulary. To realize this, the logical consequence is to build the documentation tree from scratch. This is usually done every time the sales department found a new customer for an already existing product.

The documentation of the complete system is written not only with the words used by the lead customer but the description is streamlined for the use in the lead customers system. This again avoids the possibility to show the system description to other customers. As the working over of the very detailed description is far to time consuming at a point in time when sales just about plans to visit possible new customers, sales usually uses a very rough description leading to promises the developers wouldn't have agreed to. By the time the specialists start talking, most of the extent that the next customer wishes is promised to a price and timeframe that causes severe problems. From that point on the "rowing back" of the development department starts, leading to massive irritation between sales, development and the project-manager.

This situation leads to an obvious requirement of the description on system level, saying that the requirements-specifications must be formulated in a general, customer independent, product describing way. The idea behind is a description of the product for somebody who does not know it. (A guideline for the product describing the functionality of everything that can be seen and touched).

The description of the Software leads to a different way of looking at the product. To work out a description dealing with only one product does not exploit all the opportunities a platform offers. This can be easily understood by looking at simple re-use examples concerning software. A product that is build into e.g. cars of customer X and communicates in the customers car via CAN, can use the same driver as a product B build in customer X's cars. Also the control of for example a clutch may be very similar for a clutch in product A and product B.

These simple examples show that a platform for system- and subsystem-requirements differs greatly from a platform for software-requirements. The typical way of building trees for projects that links the customer requirements to the system requirements which again are linked to the sub-system-requirements which again are linked to the software-requirements doesn't work for a platform regarding the needs of system and software at the same time anymore (see figure [different levels of requirements]).

3.2 System and Software - one Tree for each

A way to solve this problem is by building one tree for the system and another for the software. Whereas the system tree contains the functionality of the product for different customers (is product oriented) and the software-tree contains functions that can be re-used over a number of products and customers (is function oriented).

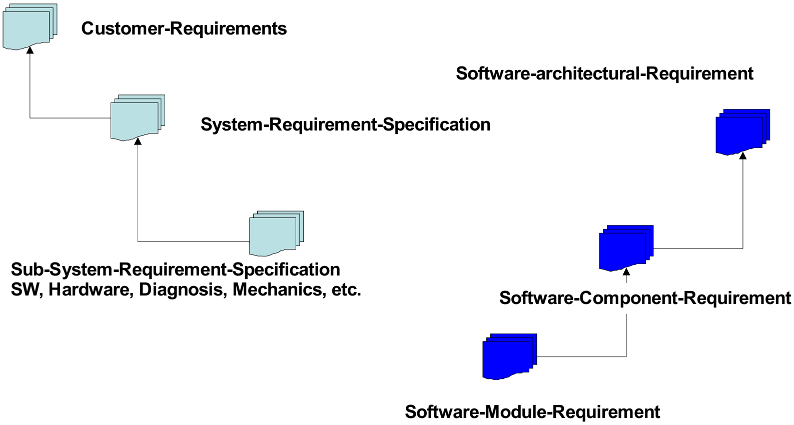

Customer-Requirements
Software-architectural-Requirement
System-Requirement-Specification
Sub-System-Requirement-Specification
SW, Hardware, Diagnosis, Mechanics, etc.
Software-Component-Requirement
Software-Module-Requirement

Figure [3] Software- and System-tree in one Figure

In order to meet the needs of the software for one product the software architecture is the crucial point. A re-use can then be established within the software-department by taking any module out of any product or any customer application and use it in a product specific architecture. A three level software-tree makes sense because of the different software components e.g. the basis software, the functional software, the interface software, etc. Therefore there must be different kinds of software components and software modules described, however only one software architecture per product. Just as well as there is only one system description but several sub-system-descriptions (e.g. mechanics, electronic hardware, diagnosis, etc.).

3.3 Different views of Software and System meet in a new framework

The different views of Software and System are shown below:

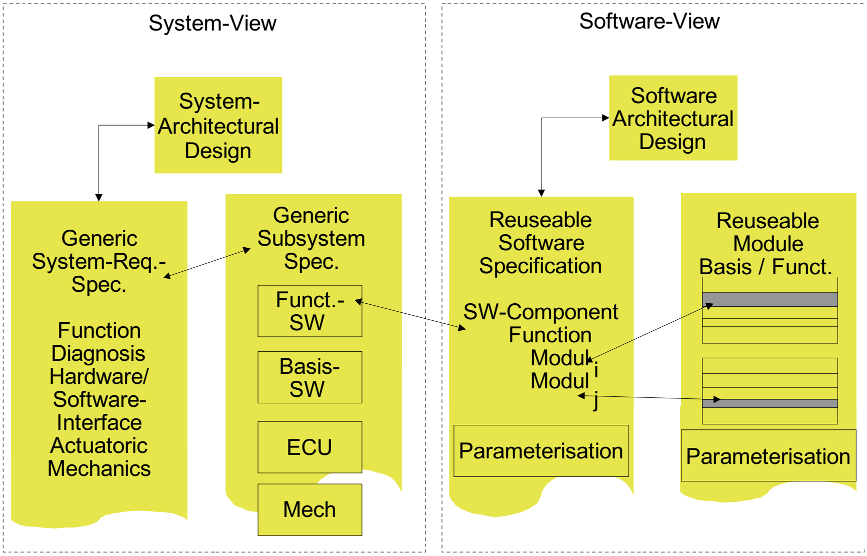


Figure [4] System- and Software-View on a Requirements Tree Structure

The arrows show the links that can be found in the trees which are build in the database. The software-modules can be re-used for different projects by parameterisation. In order to be able to build the system tree in a way that it is easy to link the requirements to the software tree, a fundamental understanding of how to describe the requirements is necessary. It is a must to start to look at the product that is described not out of a part view (or a description of what I see), but out of a purely functional point of view. In this way the thinking becomes very similar to the way the FMEAs are build. In the FMEAs the function and failure nets tend to become much more important than the links between parts. The question that needs to be answered is: What function does a specific part have?

By answering this question the description of the requirements on a system- and sub-system-level is a description of what the product is supposed to be able to do. In other words: The system requirement specification is a top level description of the functionality of the product and the sum of the sub-system requirement specifications is a low-level description of the functionality of the product.

Coming back to an earlier described problem, it is now obvious that this way the system requirement specification can be used by the sales department to introduce the capability of the product to a customer. Since it makes sense to describe the maximum of what my product can do in a platform description the sales thus has the complete capability and can directly make out any wish of the customer that goes beyond what is existent. This again leads to a very precise project management concerning development costs and timeframe.

Of course stepping down the supply chain, the same can be said about every level of negotiation, as the same is true for the specialists on the subsystem level, as well (e.g. for diagnosis and so on).

3.4 Nothing new in Software-Development?

The structure of the tree forces the software developer to program in a way that makes re-use of modules possible. The best way of doing this is to achieve the variant handling with the help of parameterisation.

Since the re-use of software as a product is easier than re-using a part out of metal there were database tools developed years ago, which have the full database ability with merging and baselining. Nevertheless the documents for the software architectural design had to be versioned outside of the database because these databases lacked the ability to include graphics. This leads to a break in the description chain and to a new baseline of the architectural design for each project. But with the help of a generic description it is very useful to baseline the software architectural design document for one product and re-use it in as many adaptation development projects using this product as possible. Because – as described earlier – the software architectural design for one product is the crucial point for the product functionality from a software point of view. It describes which software modules are used in what way to fulfil a certain sub-system functional requirement. This allows the same structure of requirements database and software-module database.

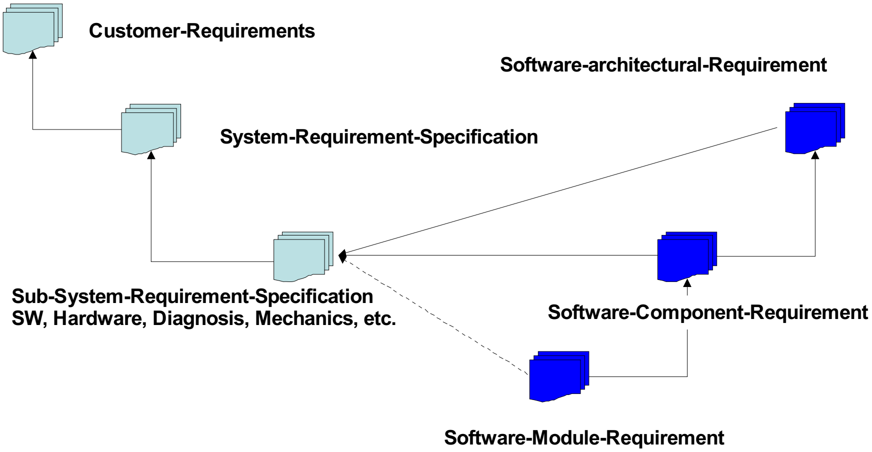

Customer-Requirements
Software-architectural-Requirement
System-Requirement-Specification
Sub-System-Requirement-Specification
SW, Hardware, Diagnosis, Mechanics, etc.
Software-Component-Requirement
Software-Module-Requirement

Figure [5] Links between the Documents

4 Testing in a Platform Project

Looking at the V-Model it is apparent that testing is a very important part in development in the SPICE-model. This can also be found when looking at the engineering processes within the frame of Automotive Spice in which the first 5 deal with the definition phase and numbers 6 to 10 with testing.

In the introduced trees for software and system the tests can be easily allocated to the different levels of definition.

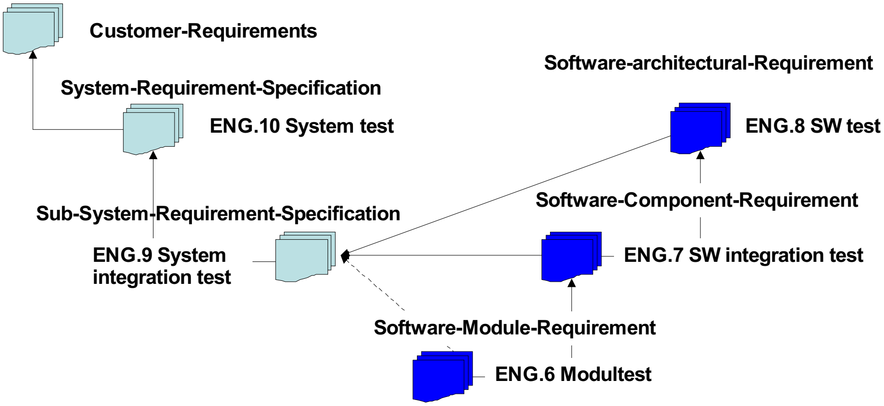

Customer-Requirements
Software-architectural-Requirement
System-Requirement-Specification
ENG.10 System test
ENG.8 SW test
Software-Component-Requirement
Sub-System-Requirement-Specification
ENG.9 System integration test
ENG.7 SW integration test
Software-Module-Requirement
ENG.6 Modultest

Figure [6] Tests in the software- and system-tree

It is not necessary to insist on ENG.7 to split between the levels of Software-Component-Requirements and Software-Architectural-Requirements. Of course it is also possible to allocate them to the Software-Components-Requirements only. The same is true for the ENG.8 white box and black box tests.

More interesting in this aspect is the connection between the definition and the test. As long as the description of each requirement is generic, it is not possible to carry out project specific tests. A generic description logically has to describe the maximum capability of the product. Therefore the tests connected must have passed the maximum requirement at least once. This way the description can be re-used as explained in the previous chapters.

Usually the different customers do not have the maximum product capability requirement in every single point. This has a high impact on every test that needs to be repeated for projects that use the platform as basis (e. g. the regression tests). To work around this problem, different fields in the database need to have different functions. The “description” field e.g. shall not be changed after being branched. The “acceptance criteria” field should be able to be changed. That is because the acceptance criteria are the basis for the test description. Each project that has lower acceptance criteria than what the product is capable of, should be able to change the tests that need to be carried out for this project to a lower passing level. For a better understanding this shall be described with an example:

A transmission that has a clutch which has been dimensioned for a torque of 600Nm may be built into a vehicle with an engine that has a maximum torque of 500Nm. The clutch controller should be tested to a maximum of 500Nm.

This example shows, that as far as the description is concerned, only one field should be changeable after the branching, whereas the tests should be able to be changed completely (usually most of the documentation can be used from the platform and only a few values need to be changed).

5 Conclusion

Although the complexity of modern mechatronic systems has reached a point at which documentation uses up a significant part of the time of the development- and test departments it is possible to reduce this documentation load by using a platform concept that is in line with the ISO 15504 standard.

By using a database tool with a frontend for each member of the development team it is possible to reach a level 2 for all required processes including the hardware development of mechanics, electronics, pneumatics or hydraulics.

By describing the requirements of software and system out of a solely functional point of view it is possible to combine the "software-tree" and the "system-tree". By building one tree within the database for the system-requirements and one for the software-requirements it can be achieved to re-use the system tree for multiple customers and one product-line. The same is true for multiple customers as well as software programs for the software tree. To reach this goal it is necessary to describe requirements out of a functional point of view in a generic way for the platform basis. This basis can be branched for every customer. The software-tree can be used with the help of parameterisation.

The software- and the system tree can be linked as shown in figure [5]. Most important is that links should exclusively be drawn from bottom up and never from top down.

Since the requirement trees are built in a similar way as the v-model, the tests which are needed according to the standard can be linked as well. The basis for the tests must always be the highest requirement whereas the branched "customer" versions "only" need to cover the customer's demand.

The often found "standard" in companies asking for an administration of software-modules and mechanical drawings separately and picking project relevant parts to combine them for each customer in order to reach the required level becomes obsolete.

6 References

[1] ISO/IEC CD 15504-5.6 Information Technology — Process Assessment — Part 5: An exemplar Process Assessment Model

[2] Messnarz, Richard; Ekert, Damjan: "Assessment Based Learning Systems –Learning from Best Projects". in: Proceedings of the EuroSPI 2006 Conference

[3] www.automotive-his.de

[4] Spork, Gunther: "Establishment of a Performance Driven Improvement Program", in: Proceedings of the EuroSPI 2007 Conference

Dr-Ing. Volker Ovi Bachmann

Dr. Volker Ovi Bachmann (info@sibac.de) is the managing director of SIBAC GmbH. He studied at Darmstadt University from 1988 till 1993 and received a degree in mechanical engineering. From 1993 till 1998 he has worked as a so-called assistant at the automotive department of Darmstadt University as researcher and lecturer and received a degree as Dr.-Ing.

Since 1998 he has been working in the automotive industry for ZF Friedrichshafen AG in several positions the latest being team manager in the quality management. He was in charge for 7 employees and the quality management of 5 large transmission development projects (with more than 60 engineers per project involved).

During his time in industry he received the qualification as a Spice Assessor. He has an experience of 120 assessment hours. In 2008 he become a competent assessor.

In Jan. 2008 he founded SIBAC GmbH. Since then he is working as a process consultant at various companies.

Dr. Richard Messnarz

Dr. Richard Messnarz (rmess@iscn.com) is the Executive Director of ISCN LTD. He studied at the University of Technology Graz and he worked as a researcher and lecturer at this University from 1991 - 1996. In 2 European mobility projects (1993 and 1994) he was involved in the foundation of ISCN, and he became the director of ISCN in 1997. He is/has been the technical director of many European projects:

- PICO - Process Improvement Combined Approach 1995 - 1998,
- Bestregit - Best RegionalTechnology Transfer, 1996 - 1999,
- TEAMWORK - Strategic Eworking Platform Development and Trial, 2001-2002,
- MediaISF - Eworking of media organisation for strategic collaboration on EU integration, 2001-2002

He is the editor of a book "Better Software Practice for Business Benefit", which has been published by IEEE (www.ieee.org) in 1999 (the leading research publisher in the USA). He is the chairman of the EuroSPI initiative and chair of the programme committee of the EuroSPI conference series.

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

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

Integrated Engineering Collaboration Skills to Drive Product Quality and Innovation

Andreas Riel[1], Serge Tichkiewitch[1]
Anca Draghici[2], George Draghici[2]
Damian Grajewski[3], Zenobia Weiss[3]
Richard Messnarz[4]

[1] G-SCOP Laboratory, Grenoble University of Technology, Grenoble, France
[2] Integrated Engineering Research Centre, Politehnica University of Timisoara, Timisoara, Romania
[3] Virtual Design and Automation Centre (VIDA), Poznan University of Technology, Poznan, Poland
[4] ISCN GmbH, Graz, Austria

Andreas.Riel@inpg.fr, Serge.Tichkiewitch@inpg.fr
adraghici@eng.upt.ro, gdraghici@eng.upt.ro
Damian.Grajewski@put.poznan.pl, Zenobia.Weiss@put.poznan.pl
rmess@iscn.com

Abstract

Collaboration skills have become an incontestable necessity of professionals in engineering teams. Engineering teams are becoming more and more interdisciplinary and distributed, products and the environments they are created, used and recycled in are increasingly complex. Design engineers, system architects as well as requirements engineers and managers have to be able to understand the product and its whole lifecycle in order to be able to respect the constraints and requirements that are imposed on the product by different actors. Many of these skills have to be developed and improved in the course of a professional career rather than in initial education. This paper gives insight into the development of one component of a consistent qualification and certification scheme which is in its entirety particularly targeted at Integrated Design Engineers in various industrial sectors.

Keywords

Engineering Collaboration, Integrated Engineering, Integrated Design, Quality Improvement, Certification, Professional Training

1 Introduction

The design of successful and sustainable products is increasingly linked to mastering the challenge of the complexity and multidisciplinary nature of modern products in an integrated fashion from the very earliest phases of product development. Design engineers are increasingly confronted with the need to master several different engineering disciplines in order to get a sufficient understanding of a product or service. Competence in the major aspects of the whole product lifecycle is a key element of the skills they require to be able to conceive a product design that fulfils the requirements of all the different actors involved in the product's lifecycle as well as the constraints imposed by their individual environments. Likewise, engineering teams are getting increasingly interdisciplinary, and thus there is a strong demand for a mutual understanding and collaboration between domain expert team members.

In this context, collaboration skills assume an increasingly important role in the development process and in the organisation. These skills reach far beyond the skills that are classically known and trained as teamworking skills, as collaboration in modern innovative product development organisations happens on inter-departmental, inter-team and inter-project levels [1]. Tools are available that support this networked collaboration, themselves requiring specialised skills to ensure their effective deployment throughout the development process.

A significant part of these integrated engineering skills are acquired throughout the professional career of engineers rather than in initial education, as they require experiences in real working environments with complex interdisciplinary development projects. Engineers thus have the need of training and certification programs allowing them to improve and certify their integrated skills along the way. Today, such internationally recognized training and certification programs for job roles in modern manufacturing do not exist. EMIRAcle [2] has partnered up with the ECQA [3] in order to define and establish job roles, curricula and certifications in the domain of Integrated Engineering on a European level. The target is to define and describe the skills that characterise Integrated Engineering in order to provide skill-specific training modules and the corresponding training material, as well as internationally recognized certification.

This paper focuses on the particular role of collaboration skills of engineers in complex engineering environments. Section 2 establishes the context with the Integrated Design Engineer job role development. Section 3 discusses the national survey based approach to establish the basis of a set describing the skills demanded in collaborative distributed engineering with a focus on design engineering. Section 4 discusses the need of modern integrated engineering qualification and certification to take into account skills of using Virtual Technologies to support collaboration. This is highly innovative in the sense that up until recently these technologies have mainly been used as support of product development on a management and coordination level.

2 Integrated Design Skill Set Development

Based on the demands of numerous industrial sectors, e.g. [4], a project team composed of members of EMIRAcle and the ECQA establishes a qualification and certification scheme for Integrated Design Engineers in the iDesigner project [5]. The relevance of this work to integrated and competitive aspects of other engineering domains—and thus also to EuroSPI²—has been shown e.g. in [6]. Figure 1 gives an overview of the highest level of skills, the so-called skill units [7] of this modern job role.

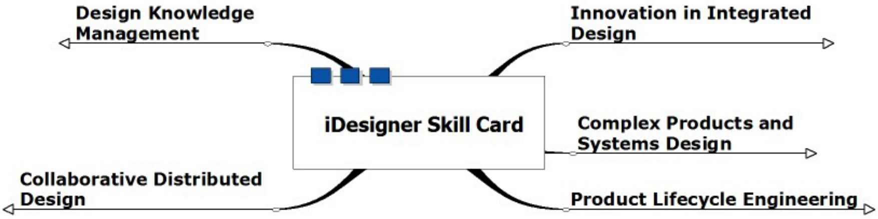

Design Knowledge Management
Innovation in Integrated Design
iDesigner Skill Card
Complex Products and Systems Design
Collaborative Distributed Design
Product Lifecycle Engineering

Figure 1. The Top-level Skill Set Definition of Integrated Design Engineers

For each of these skill units so-called skill elements have been specified, which give details on the skill requirements, and thus provide the framework for the development of training modules and test questions for the certification.

- The skill unit “Innovation in Integrated Design” is about driving innovation creatively and systematically in design.
- “Complex Products and Systems Design” addresses the key skills that are required from Integrated Design Engineers to be able to tackle the complexity of modern products and systems and the environments they are created and operated in.

- “Product Lifecycle Engineering” is about the skills required to be able to take into account the whole lifecycle of a product in terms of the constraints and requirements imposed on the product by its different involved actors.
- “Design Knowledge Management” addresses the formalization and sharing of design knowledge, as well as the capitalization on such knowledge.
- “Collaborative Distributed Design” deals with key skills required for engineers to be able to work in distributed and multidisciplinary teams.

The following section 3 is dedicated to the latter set, in that it points out an approach to identify and formalize collaboration skills in engineering teams. Section 4 deals with the use of Virtual Technologies to support effective engineering collaboration, which is a subject of the “Complex Products and Systems Design” unit. In [8] we have introduced our work from the point of view of the “Product Lifecycle Engineering” skill unit.

3 Collaborative Distributed Design

This section shows a practical approach for collecting and processing key information relevant for the specification of the Collaborative Distributed Design (CDD) skill unit, which is part of the full skill set of an Integrated Design Engineer, and it is related the particular needs of the engineer to fulfil his professional duties with respect to precise performance criteria.

3.1 Approach to Skill Requirements Identification

The approach presented here has been implemented by a member team involved in the iDesigner project [5] in order to collect as much and diverse relevant information as possible to define the skill unit consistently and universally. Figure 2 shows the positioning and the motivation of these preliminary researches in the context of the CDD skill unit definition.

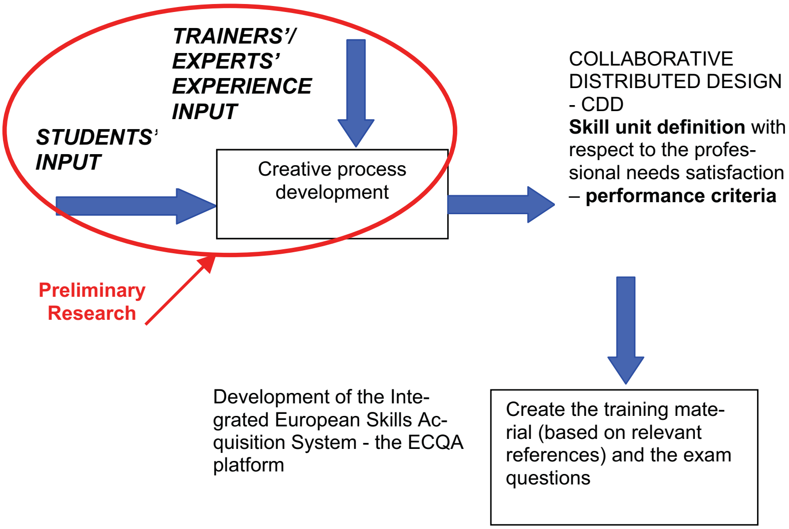

TRAINERS'/ EXPERTS' EXPERIENCE INPUT
STUDENTS' INPUT
Creative process development
COLLABORATIVE DISTRIBUTED DESIGN - CDD
Skill unit definition with respect to the professional needs satisfaction – **performance criteria**
Preliminary Research
Development of the Integrated European Skills Acquisition System - the ECQA platform
Create the training material (based on relevant references) and the exam questions

Figure 2. The Process of the Development of the Skill Unit CDD

The research scenario and the development plan consist of the steps which are outlined in the sections below.

3.1.1 Experience Input by Trainers/Experts

The project team members have established a preliminary structure of the CDD skill unit on the basis of the results of a brainstorming session and expert discussions concerning the skill unit's practical aspects in the whole project consortium.

By analyzing the scenario of integrated design team work, experts have concluded that a key characteristic of this working process is linked with the human relationship development, which is why they have considered the need of a specific skill unit named Collaborative Distributed Design (CDD). This is linked with designers' behaviour oriented towards attaining the design process objectives, which is the final design solution being well accepted by each actor involved in the process that aims at developing and producing products that satisfy customers' needs in an optimal way.

In the course of this process, the trainers/experts have identified the following main skill elements:

- *Design Process Moderation*,
- *Working in Distributed Engineering Teams*,
- *Communication with experts from different fields*.

3.1.2 Input by Students working in Industry

The next step consists of collecting information from people representing a target group of the iDesigner project, i.e., the end users of the qualification and certification related to his/her specific needs as actors/designers in integrated design teams. This preliminary study was conducted by the "Politehnica" University of Timisoara, Romania as the team member of the iDesigner project which is in charge of the CDD skill unit development.

The research sample consists of 21 master students (first year of study) from the Integrated Engineering specialization who are at the same time all employees of some prestigious companies from the Western part of Romania (Timis country), like: CFR, Valeo, Flextronics, Alcatel, HNP, Aton, Moreno, Hella and Draexlmaier.

The research methodology is a qualitative one, based on group phenomenological analysis and a technique of non-directly centred group interview parallel with the questionnaire technique. The members of the research sample have been asked to express their professional needs regarding the three skill elements that were established in the first step of the research. All of them had experience from discussions in numerous design sessions in their companies, and they were encouraged to give their opinions and share their experiences.

Table 1 shows the results of the creative session. The items chosen for the "practical needs" were correlated with the questions of the interview. For all the identified practical needs the importance rate corresponding to the degrees $a_{ij}\varepsilon\{0,1,2,3\}$ has been calculated, corresponding to the following scale:

- 0 – needs of no importance;
- 1 – needs of little importance;
- 2 – important needs;
- 3 – very important needs.

For the interpretation of the research results, the following figures have been calculated:

The mean value: $$V(D_i)=\sum_{j=1}^{n} a_{ij}\ , \qquad (1)$$

where i=$1\div m$ is the number of items lines or practical needs, and j=$1\div n$ is the index associated with the number of subjects.

The mean average: $(D_i) = \frac{V(D_i)}{n} = \frac{1}{n}\sum_{j=1}^{n} a_{ij}$ (2)

The research results have been influenced by the fact that all the persons of the sample were young engineers who had graduated in 2008 and had their first working experiences in different companies. Most of their comments were related to the missing education aspects in the university curricula. They all emphasized important issues they were confronted with in practical engineering situations, which they were unable to master due to lack of education.

Table 1: Students' Input Information - Research Results

Skill Element	**Practical Needs (future competence prove to attend the performance criteria)**	**D_i – Degree of importance**
Design process moderation	Negotiation ability	0.929
	Ability of design activity coordination and monitor – moderation abilities	1.786
	Ability for design solutions' argumentation	0.981
	Analysis and synthesis as systematic approach	0.529
	Ability of information interpretation and formalization	0.586
	Constructive critics as a way of thinking	0.214
	Self-control capacity (patience)	0.257
	Diplomacy and flexibility	0.216
	Creativity	0.957
	Ability to have a global assembly vision of the design project	0.916
	Ability to use the team working techniques - Inter-relationship motivation and group debate for design process	0.916
Working in distributed engineering teams	Ability to operate with different tools for collaborative design	1.000
	Flexibility and diplomacy	1.375
	Hard working ability and under the stress conditions	1.563
	Ability to operate and to work in a design team of specialists (team working techniques)	1.063
	Initiative and creativity	1.133
Communication with experts from different fields of activity	Ability to communicate in face-to-face meetings (verbal and non-verbal communication)	1.133
	Communication using a simple language	1.267
	Ability to organize and to identify the relevant information	1.400
	Ability to communicate with specialists from different fields of sciences (find the common language for better understanding one each other)	0.733
	Ability to communicate with specialists from different cultures (avoid misunderstanding, miscommunication, constructive critics and awareness etc.)	0.714

3.2 Specification of the Skill Unit

Based on the results of this study and previous research and investigations, an initial design of the skill unit has been established for discussion in the whole project consortium. It turned out that all the consortium partners (from four different countries in Europe) shared the same experiences.

By consequence the following skill units and elements have been agreed upon:

1. *Design Process Moderation*, with the following skills elements:

- particularities of negotiation in the design process;
- argumentation and moderation;
- representation of the design solution for different actors' views;
- team working (working group techniques, inter-relationship motivation and group discussions for the design process);
- knowledge engineering (knowledge search, design database operations).

2. *Working in Distributed Engineering Teams*, with the following skills elements:
   - human interaction in distributed engineering teams;
   - tools for collaborative design (Skype, AREL, videoconference systems, WebEX etc.).

3. *Communication with Experts from Different Fields* with the following skills elements:
   - Team Communication;
   - inter-cultural communication particularities;
   - multicultural skills development.

The consortium believes that this skill set largely corresponds to the one required in the Software Design Engineering domain, since none of these skill elements is relevant for only a specific engineering sector.

4 Augmented Design Collaboration

Whereas the skill set developed above addresses mainly traditional aspects of distributed collaborative design, the complexity of modern products and the environments they are conceived, produced and operated in demand the use of modern so-called Virtual Technologies (VT) to support and leverage the collaborative design process. The following sections outline scenarios that serve as guidelines for the development of the corresponding Designer skills elements in the iDesigner project.

4.1 Virtual Technologies for Collaborative Design

The interest of using Virtual Technologies in the context of Integrated Design is to enable all the actors involved in the design process to spend more time and efforts effectively in the problem space of design projects rather than in the solution space. The major Virtual Technologies addressed in this context are the following:

- Virtual (VR) and Augmented Reality (AR) Environments
- 3D Holographic Tables and Displays
- Reverse Engineering (RE)
- Rapid Prototyping (RP)

VR and AR environments, as well as 3D holographic tables allow all the actors involved in the lifecycle of a product to experience the product and the environments that this product will be operated in before the product and its environments actually exist. The main enablers of such systems are hardware and software allowing the simulation of all the properties of the product and the environments that are significant for the targeted users. If they are used at all, VR and AR environments are mostly used as support on project and program management levels rather than as support for engineers.

RE techniques and devices like 3D scanners allow to feed the virtual worlds constructed in VR and AR with 3D models from real objects. According to [9] prototyping is important for designers and all actors in the Product Lifecycle. On the other hand, however, building prototypes is generally considered too expensive in industry. Rapid Prototyping techniques and devices allow the quick and cheap production of prototypes in sufficient quality to support decisions in design processes. They thus provide a way for designers to realize their models constructed in the virtual world in a cheap and immediate way.

During the design process, designers have to define the structure of their product while considering its functional definition. This design phase remains little assisted for designers. In addition, as numerous options have to be considered for the end of life of a product (reuse, remanufacturing, recycling, etc.) it becomes more difficult to obtain a compromise concerning the final structure of the product. Product-Service Co-Design [10] is also leveraged by putting designers into usage environments (make them live and carry out use cases themselves). The use of VR and AR can be an effective support for designers to transform the functional definition of the product into the design of its structure during the conceptual design phase [11], consequently helping to improve the final product quality and to reduce error rates [12]. Methods and techniques allowing the integration of VR as a user interface into the process of geometric modelling and detailing have been shown and implemented [13].

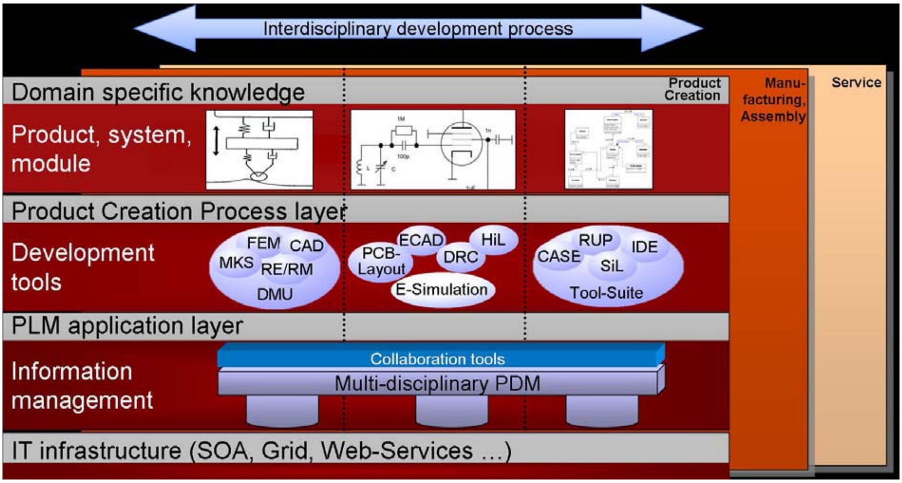


Figure 3. Dimensions of Product Creation with Virtual Technologies [14]

Figure 3 from [14] shows the various dimensions of modern product development with the support of Virtual Technologies. The basic IT infrastructure is the foundation for the inter-departmental and inter-supplier information management and collaboration tools. We have reported about the involvement of Integrated Design Engineers in the use of these tools e.g. in [6]. On top of those, different kinds of CAE, modelling and simulation tools are applied in the mechanical, electrics/electronics, and software domains. Those tools and models provide the building blocks for the product level. In the third dimension, Manufacturing, Assembly, Services, Disposal and Recycling – short, all phases of the product life cycle – are linked to the development in the following respects:

- They impose requirements and constraints to development.
- They give essential feedback information required to innovate the product incrementally or radically.
- Simulation models and calculation results from development can be re-used and/or validated and thus improved in those phases of the lifecycle, where the real product exists.

This subject is further developed in the context of front-loading development tasks by simulation and Virtual Technologies in [15]. In terms of the subject developed in this paper, it is important to understand that Design Engineers assume a crucial role in this complex process, as they

- have to take into account the constraints and requirements imposed by all the actors in all lifecycle phases in their design models;
- provide the basis for the simulation models developed in other phases.

Design engineers who lack integrated design skills are unable to act as linking elements in the multi-dimensional product creation environment in Figure 3.

5 Conclusion

This paper presents the requirements of collaboration skills of engineers, with a particular focus on Design Engineers. It does this in the context of a project that aims at the development of a skill set for the qualification and certification of Integrated Design Engineers on a European level. In the first part it describes the process of the identification of an initial set of skill elements on the basis of a survey done with a number of young Romanian graduates with experience in international enterprises.

The second part is devoted to the use of advanced Virtual Technologies to support collaboration in design, again in the context of skill identification. It is shown that Design Engineers without basic skills in using such virtual technologies to support collaborative design have difficulties in tackling the complexity of design tasks linked to modern products. We conclude that in addition to the qualification of Integrated Design Engineers in distributed collaboration skills, the qualification in the use of Virtual Technologies is indispensable in modern product development. As these tools and associated skills are primarily targeted at improving the understanding of the whole product from different actors' points of view, we consider them also highly relevant for all engineers who are supposed to have a holistic, integrated view on complex systems, e.g. systems architects and requirements engineers and managers.

Acknowledgements

This research is currently supported by the EU in the Leonardo da Vinci projects LLP-1-2007-AT-KA3-KA3MP (EU Cert - EU Certification Campus) and [5] of the Lifelong Learning Programme. The launching activities have been supported by the European Commission under the contract NMP2-CT-2004-507487 of the Network of Excellence in FP6 VRL-KCiP, which has turned into the EMIRAcle association [2] in June 2008. In this research EMIRAcle collaborates with the ECQA [3], which has been created based on the results of numerous successful projects in the Leonardo da Vinci programme.

Literature

1. Messnarz R., Spork G., Riel A., Tichkiewitch S., Dynamic Learning Organisations Supporting Knowledge Creation for Competitive and Integrated Product Design. In: Proceedings of the International CIRP Design Conference, Cranfield, March 2009
2. EMIRAcle – European Manufacturing and Innovation Research Association, a cluster leading excellence – www.emiracle.eu
3. ECQA – the European Certificates and Qualification Association – www.ecqa.org
4. Menne R., Ford: teach integrated engineering, in: Automotive Engineer, December 2007, Professional Engineering Publishing Limited, London, UK, p. 5, 2007
5. iDesigner – Certified Integrated Design Engineer, Leonardo da Vinci contract LLP-LdV-TOI-2008-FR-117025
6. Riel A., Tichkiewitch S., Messnarz R., Integrated Engineering Skills: Improving your System Competence Level. In: EuroSPI 2008 Industrial Proceedings, Dublin, Delta Series about Process Improvement, ISBN 978-87-7398-150-4pp. 8.9-8.20.
7. Messnarz R. et al., The EQN Guide. Graz, Austria, 2008
8. Riel A., Tichkiewitch S., Grajewski D., Weiss Z., Draghici A., Draghici G., Messnarz R., Qualification and Certification of Life Cycle Engineering Skills of Design Engineers. In: Proceedings of the CIRP Lifecycle Engineering Conference 2009, Cairo, to appear
9. Brown T., Design Thinking. In: Harvard Business Review June 2008, pp. 84-96

10. Schuh G., Gudergan G., Service Engineering as an Approach to Designing Industrial Product Service Systems. In: Proceedings of the 1st IPS² Conference, Cranfield, 2009, ISBN 978-0-9557436-5-8, pp. 1-7

11. Zwolinski P., Tichkiewitch S., Sghaier A., The Use of Virtual Reality Techniques during the Design Process: from the Functional Definition of the Product to the Design of its Structure. In: Annals of the CIRP Vol. 56/1 (2007) : 135-138

12. Neugebauer, R., Weidlich D., Zickner, H., Virtual Reality Solutions for the Design of Machine Tools in Practice, Proceedings of the 5th CIRP International Seminar ICME, 2006, 88-95028-01-5

13. Weidlich D., Cser L., Polzin T., Cristiano D., Zickner H., Virtual Reality Approaches for Immersive Design In: Annals of the CIRP Vol. 56/1 (2007) : 139-142

14. Stark R., Krause F.-L., Kind C., Rothenburg U., Muller P., Stockert H., Competing in Engineering Design – the Role of Virtual Product Creation. In: Proceedings of the 19th CIRP Design Conference, Cranfield, 2009, ISBN 978-0-9557436-4-1, pp. 1-8

15. Riel A., From DMU to BMU: Towards Simulation-Guided Automotive Powertrain Development. PhD Thesis, Vienna University of Technology, Vienna, Austria, 2005

Author CVs

Andreas Riel

Doctor Andreas Riel is General Manager of the EMIRAcle association, and is mainly responsible for external communication and transfer to industry. He has ten years of professional work experience in the automotive industry, and two years of experience in the management of the European Network of Excellence VRL-KCiP, which is at the origin of EMIRAcle. Over the recent years he has gained vast experience in innovation management in both product development and manufacturing, and he has been certified as European Innovation Manager and Trainer. He is the coordinator of the iDesigner project [5]; Andreas.Riel@inpg.fr

Anca Draghici

Professor Anca Draghici of the Polytechnical University of Timisoara has a long scientific experience in the fields of entrepreneurial behaviour and strategies, human resources management, and knowledge management. She is mainly involved in research projects related to collaborative design, ergonomics design and e-platform for entrepreneurship evaluation; adraghici@eng.upt.ro

George Draghici

Professor George Draghici of the Polytechnical University of Timisoara has a vast academic and scientific experience in integrated engineering and he is the director of the Integrated Engineering Research Centre (training and consulting for SMEs). He is also the director of the Romanian INPRO network of academic and industrial institutions; gdraghici@eng.upt.ro

Damian Grajewski

Damian Grajewski is a PhD student at the VIDA (Virtual Design and Automation) Centre at Poznan University of Technology. He has a long and profound experience in the application of Virtual Technologies in industry-driven research projects; Damian.Grajewski@put.poznan.pl

Zenobia Weiss

Professor Zenobia Weiss of Poznan University of Technology is the founder and director of the Virtual Design and Automation (VIDA) Centre at the same university. She is a world-renowned scientist in the area of Virtual Technologies; Zenobia.Weiss@put.poznan.pl

Serge Tichkiewitch

Professor Serge Tichkiewitch is currently the Scientific Coordinator of the Network of Excellence "Virtual Research Lab for a Knowledge Community in Production" and has been elected as the President of the EMIRAcle association. He has been the former Director of the Industrial Engineering and Management School of Engineers at National Polytechnic Institute of Grenoble. He was the initiator and director of the Integrated Design team of the 3S Laboratory, composed of 40 persons and the first General Director and founder of the French PRIMECA network. He has supported 20 PhD thesis, published more than 130 papers in journals, books or proceedings of international conferences. His research interest is the methodologies and the tools for integrated and innovative design and the modelling of the forging process; Serge.Tichkiewitch@inpg.fr

Richard Messnarz

Doctor Richard Messnarz has 16 years industry experience and is the chairman of EuroSPI, the coordinator of the EU Certified Innovation manager, the coordinator of the European Qualification Network, the moderator of task forces of 24 leading German companies, the editor of 7 books (IEEE, Wiley, Springer) on process improvement, and the director of ISCN since 1997. He is a principal ISO 15504 assessor working for firms like ZF Friedrichshafen AG, Magna, Continental, T-Systems, Giesecke & Devrient, etc.; rmess@iscn.com

A Systems Approach to Software Process Improvement in Small Organisations

Diana Kirk and Stephen MacDonell

Abstract

There is, at the present time, no model to effectively support context-aware process change in small software organisations. The assessment reference models, for example, SPICE and CMMI, provide a tool for identifying gaps with best practice, but do not take into account group culture and environment, and do not help with prioritisation. These approaches thus do not support the many small software organisations that need to make effective changes that are linked to business objectives in short time periods. In this paper, we propose a model to support such change. We base the model on an analogy of 'software system as human' and suggest that we can apply the idea of human health to help identify business objectives and improvement steps appropriate for these objectives. We describe a 'proof-of-concept' case study in which the model is retrospectively applied to a process improvement effort with a local software group.

Keywords

Software process improvement, process modelling, systems approach

1 Introduction

Existing models for software process improvement (SPI), for example, CMMI (Chrissis et al. 2007), ISO/IEC 15504 (SPICE) (ISO 2006) and ISO/IEC 12207 (ISO 1997), have been criticised by researchers and practitioners as being limited to large, traditional organisations and failing to provide the necessary guidance for small software groups (CaterSteel 2001, Grunbacher 1997, Huack et al. 2008, McCaffery et al. 2008). As software groups comprising fewer than 25 persons represent a majority in Europe, Ireland, Canada, Brazil and elsewhere (Laporte et al. 2008), this is seen as a major issue. Some characteristics of smaller organisations that may affect the success of SPI adoption include more informal management and planning, greater need for flexibility, a human-centric culture (Laporte et al. 2008), 'flatter' structure with imprecisely-defined responsibilities, a lack of exposure to standards and limited funds (Grunbacher 1997). Such characteristics imply an environment where dependence upon individuals is high. It has also been observed that existing models state which processes should be in place, but provide no guidance as to which to implement first (Chen et al. 2008). This means that the need of small organisations to prioritise according to business objectives (Aaen 2003, Laporte et al. 2008) is not supported.

We have earlier suggested that a model to support activity selection during software projects must take into account the business-related objectives for the software project and have proposed that a more holistic and flexible approach to process selection involves a change in focus from 'defining activities' to 'selecting activities to meet objectives' (Kirk 2007). In this approach, focus is on the whole system i.e. is not limited to considerations of cost and quality but rather includes consideration of human-related factors. For example, the owner of a small software organisation may be extremely interested in retaining and increasing the knowledge of developers and so may consider 'developer knowledge' to be of importance. This understanding may inform his choice of process and he may, for example, choose informal reviews over unit testing as a means of meeting objectives.

The need to focus on system objectives during software process improvement (SPI) initiatives is also suggested by others. Aaen (2003) criticises the use of existing assessment models as creating "a blueprint of a future software process" without providing any understanding of how processes emerge. He believes that it is necessary to understand an organisation's values and goals before understanding how it may change and that a preferred approach would be to support process users in deciding what a specific situation requires. Laporte et al. (2008) have found that one reason for the failure of small organisations to adopt standard models is that such organisations "find it difficult to relate ISO/IEC 12207 to their business needs".

We have suggested that a fruitful analogy to aid understanding of contextualised software systems is to consider the software system as a human and to apply ideas from human health (MacDonell et al. 2008). With this analogy, the focus is on identifying gaps in values of relevant 'health' factors and selecting activities to close the gaps. In this paper, we extend this analogy and apply it as a basis for a model for SPI. We explore the potential usefulness of the model by retrospectively applying it to an SPI initiative with a small, local software group. We then show how we have used the model as a basis for creating hypotheses for more formal investigation in small software organisations.

2 Related work

There is increasing interest in supporting software process improvement in small groups as a result of the realisation that such groups form a majority in many countries (Laporte et al. 2008). A selection from the literature is presented below.

The International Standards Organisation (ISO) has established a working group to address the creation of a software engineering standard tailored to very small enterprises (Laporte et al. 2008). The approach taken is to tailor an existing Mexican standard, *MoProsoft*, for small and medium enterprises. MoProsoft is based on ISO/IEC 12207, with practices from ISO9001, CMMI, the Project Management Body of Knowledge and the Software Engineering Body of Knowledge.

The University of Southern Queensland has developed a method, *RAPID* for software process assessment in small organisations and have applied this method in four organisations (CaterSteel 2001). The approach involved selecting eight processes based on ISO/IEC 15504 and restricting assessment to rating levels 1-3.

McCaffery et al. (2008) introduce *AHAA*, a "new low-overhead method that has been designed for small-to-medium-sized organisations wishing to be automative software suppliers". The method integrates the structured-ness of CMMI and Automative SPICE with the flexibility of agile practices. The development of AHAA included a restriction of the CMMI process areas most suitable for inclusion in an SPI model for small-to-medium-sized organisations, based on a number of criteria extracted from the literature. The four process areas selected for the first release were Requirements Management, Project Planning, Project Monitoring and Control and Configuration Management.

Pikkarainen et al (2005) discuss deploying agile practices in organisations and applies a framework based on a continuous improvement ideology that "addresses the importance of utilizing the experiences of the software developers" as an important input to SPI. The approach involves selecting the agile practices to be deployed and the author comments that the "existing ways to discover the agile methods to deploy are unstructured" (Pikkarainen et al. 2005).

In the above examples, the approach is to select a sub-set of process areas from established models and create assessment models based on this subset. The resulting models have been applied with some success. However, none supports the ability to choose a project-specific development model based upon key objectives or to make trade-offs when planning changes (MacCormack et al. 2003).

3 Human health analogy

We have proposed that a useful analogy to aid understanding of software system health is that of the human system. In this Section, we expand on this analogy in order to provide a motivation for our SPI model. Some drivers of the analogy are (MacDonell et al. 2008):

- Human health is established by measurement of indicators, for example, blood pressure and cholesterol levels. We measure the 'health' of a software system (in its broadest sense, as described in Section 4) by indicators such as cost and defect levels and stakeholder satisfaction.
- Humans pass through a number of life stages, for example, adolescence and mid-life crisis. Each stage exhibits some common characteristics. For example, mid-life crisis might occur when the children leave home and 'business as usual' is no longer appropriate, forcing a struggle to fit in with new situations and expectations. Software systems may also be perceived as having similar 'life stages'. For example, a step change in technology may result in an established software product no longer behaving as required, forcing efforts to make the product 'fit in'.
- The relevant indicators for humans and their expected values depend upon the life stage. For example, an Apgar test is carried out on newborn babies to establish health; the 'normal' pulse rate for an infant is different from the 'normal' rate for an adult. In a similar way, for a software system it is expected that the numbers of defects identified when the system is 'in adulthood' (i.e. established in the field) will be far fewer than when the system is 'in embryo' (under development).
- Human health is dependent upon environmental factors. For example, a thin person may be 'healthy' in a hot country with food freely available but may not fare so well in a very cold climate with low food availability. In a similar way, software targetted for experienced users may cease to be 'healthy' when the customer base extends to include naive users.
- Human health can be affected by behaviours. For example, mothers can support a positive outcome for babies by eating well and not smoking. Software systems can also be affected by behaviours. For example, developers can support a positive outcome by following best practices.
- Once a human becomes unhealthy (as defined by indicators such as blood pressure), considerable effort is required to return to health. Success depends upon the human's willingness to change behaviours and the availability of opportunities to effect the new behaviours. For 'un-

healthy' software systems, considerable effort is also required to effect change as factors such as cost and resistance-to-change come into effect.

We observe that a human may embark upon a health improvement initiative for one of a number of reasons (MacDonell et al. 2008):

- Sickness. The person may be experiencing symptoms that indicate sickess, for example, chest pains or headaches. The physician will probably check a number of key indicators, for example, blood pressure and temperature, for values that deviate from 'normal'. As a result of findings, the physician will infer the root cause of the symptoms and suggest a treatment that will remove the root cause, thus returning indicator values to 'normal' and removing symptoms. During diagnosis of root cause, the physician will probably take into account the specific life stage of the person. The suggested treatment must a) take into account the human system in a holistic way and b) consider the constraints imposed by contexts. For example, medication that lowers blood pressure but induces depression is probably not an ideal solution; nor is medication that lowers blood pressure for a person who reliably fails to take prescribed medication.
- Prevention. The person may choose to monitor health in a proactive way, for example, undergo a yearly check of cholesterol levels and blood pressure. Should values be abnormal, the physician will generally progress as for 'sickness'.
- Growth. The person may have some goal that involves improving physical or mental capability, for example, 'run a marathon'. In this case, the first task is to identify appropriate indicators and the changes required, for example, 'increase stamina' and the next task is to choose a suitable activity that addresses required changes, for example, 'running'. Again, choosing a suitable activity involves both considering indicators in a holistic way and identifying factors that may affect success. For example, if I live on a busy street and lack motivation, I may decide that 'personal trainer at the gym' will give me a better chance of success than 'running a circuit from my home at 5:30 a.m.'. However, if I am concerned about financial status, a personal trainer might be too expensive and I might decide to join a group fitness class instead.
- Adaptation. The person may be required to move to a new environment, for example, leave the childhood home or move to a different country. Behaviours that worked well in the original environment, for example, leaving cooking to others or speaking in English, may be ineffective in the new one. To mitigate the risk of failure-to-adapt, (s)he must identify the gap in key indicators (for example, `independence') and aim to close the gap by suitable activity selection.

In Figure 1, we illustrate the analogy with an example for each motivation (MacDonell et al. 2008).

Scenario	**Person**	**Software**
Sickness		
- Symptoms	Headache	Customers unhappy
- Indicators unhealthy	Blood pressure	Defect numbers
- Find cause and treat	Take medicine	Requirements process
Prevention		
- Monitor indicators	Cholesterol, lipids	Defect levels
- Preventative action	Lifestyle change	Process/product change
Growth		
- Identify objectives	Run a marathon	New innovative product
- Confounding factors	Motivation	Processes don’t support creativity
- Make changes	Training, diet	Gap analysis and change
Adaptation		
- New environment	Redundancy	Business environment
- Indicators gaps	Computer skills	All web-based
- Confounding factors	Confidence	No web expertise
- Close gaps	Computer course	Hire web developers

Figure 1: Human and software systems: SPI examples

A key observation from the analogy is that, rather than focussing on processes, as is common for SPI models, we focus on goals and indicators and it is the indicator values that inform processes. Relevant indicators are situation-specific and thus appropriate process is situation-specific. For example, if I want to improve my ability to speak French, I do not need to improve my cholesterol level.

In the next Section, we introduce a model for SPI based on an extension of the above analogy.

4 Proposed model

We commence our description of the proposed model by overviewing the architecture of a software system from the perspective of our analogy. A software system comprises a number of components (see Figure 2) and associated with each of these is a number of representative characteristics.

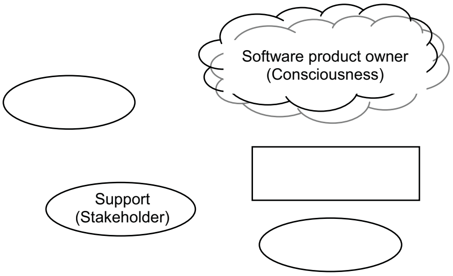


Figure 2: Software system components

- *Software product.* The 'body' of the software system is the software product or products. Common characteristics include quality indicators, such as defect density, cost indicators, such as effort, and content indicators, such as number of features.
- *Software product owner.* The ' consciousness' of the software system is represented by the entity that has authority for making decisions about planned change to the *software product* or its *stakeholders*, generally the organisation responsible for creating and deploying the software product(s). Common characteristics include organisational maturity, size, culture and management style.
- *Stakeholders.* The environment for the software system includes all humans with an interest in the *software product*. These may include members of the development organisation (for example, developers, project management, QA and support personnel) and the deployment organisation (for example, purchasers and users). Stakeholders may effect unplanned change to the product environment. For example, experienced users of a product may be replaced by inexperienced users. Common characteristics relate to skills, experience and personality type.

We now extend the ideas from Section 3 to create a methodology for analysing the health of software organisations and recommending change. We begin with some definitions:

- **Software system**. Comprises the *software product*, the *software product owner* and *stakeholders*.
- **Key indicators**. Factors that characterise the various components of the *software system* and are identified as being relevant for a specific SPI initiative.
- **Goal indicators.** *Key indicators* that represent the desired level of health of a software system, for example, relating to cost, quality and satisfaction levels.
- **Context indicators**. *Key indicators* that characterise the software system's ability to change the values of *goal indicators*, for example, relating to cost, motivation and skill levels.
- **Software system lifecycle**. The stages through which a software system passes, for example, 'Childhood' and 'Adolescence', as defined in MacDonell et al. (2008). Each stage is associated with changes to some *key indicators*, for example, 'adolescence' is associated with high levels of defects discovered in the field.
- **Symptom.** Problem reported by any *stakeholder*.

- **Prevention.** An assessment requested by the *software product owner* in which no *symptoms* are reported, rather the need is to 'check that everything is fine'. The assessment will result in a categorisation of the *software system* as one of *sickness*, *growth*, *adaptation* or *health*.
- **Growth.** Planned change to *product* or *product owner* that is outside 'business as usual', resulting in a gap between current and desired *goal indicator* values. For example, a plan for an innovative new product may mean that developer and test expertise becomes 'low'.
- **Adaptation.** Unplanned change to *stakeholders* also results in a gap between current and desired *goal indicator* values. For example, if naive users are permitted to use a product intended for use by experienced users, the 'product usability' level will fall.
- **Sickness.** Values of *goal indicators* are lower than expected for one or more of *software product*, *software product owner* or *stakeholders* and the *software system* is not in *growth* or *adaptation*. For example, effort or defect numbers may be too high or satisfaction levels too low.

Application of the model involves carrying out the following three steps (see Figure 3). For each, we present some examples to illustrate the need to take into account *context indicators* and *software system lifecycle* stages.

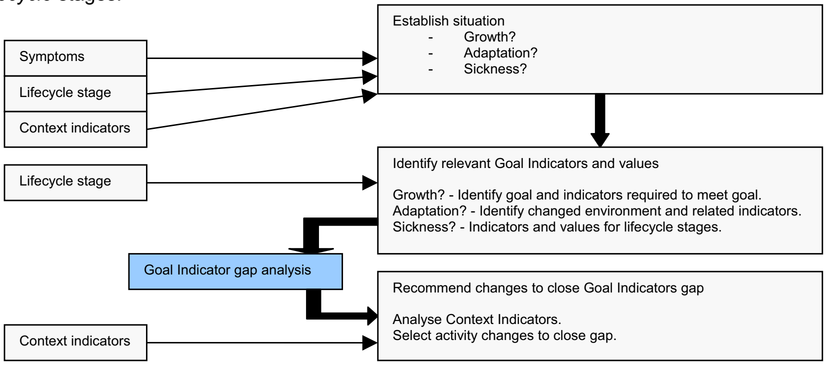

Symptoms
Lifecycle stage
Context indicators
Establish situation
- Growth?
- Adaptation?
- Sickness?
Lifecycle stage
Identify relevant Goal Indicators and values
Growth? - Identify goal and indicators required to meet goal.
Adaptation? - Identify changed environment and related indicators.
Sickness? - Indicators and values for lifecycle stages.
Goal Indicator gap analysis
Context indicators
Recommend changes to close Goal Indicators gap
Analyse Context Indicators.
Select activity changes to close gap.

Figure 3: SPI model steps

4.1 Step 1: Establish if growth, adaptation or sickness

We first interview staff to establish whether the initiative relates to a situation of *growth*, *adaptation* or *sickness*. We take this approach also in the case of a *preventative* assessment initiative.

For *growth*, we look for 'business-not-as-usual'. For example, a medium-sized organisation, A, is 'doing well', with a mature product sold to a global market (*adulthood*). Although management reports some existing problems with quality, we learn that there are plans to launch a new, innovative product into a marketplace characterised by rapidly changing technology. We categorise as a *growth* situation.

A mature organisation, *B*, with an established product used by experienced personnel would like an assessment to 'check things out' (*prevention*). When interviewing members of the support team, we discover that an increasing number of issues are being logged by users who 'do not know how to use the product'. Further probing with management reveals that downsizing in the client sector has resulted in the product being used by 'naive' users. The situation is one of *adaptation*.

Organisation C is a small group with low levels of formal process and reports problems of product quality (*symptoms*). In the absence of *growth* or *adaptation* scenarios, we categorise as *sickness*.

4.2 Step 2: Identify goal indicators and establish gap

We next establish the business objectives of interest within the given situation. We use these to help inform *goal indicators* and establish gaps between desired and current values.

Goals for organisation A relate to timely delivery and marketing and selected *goal indicators* are ‘time to market’ and ‘number of hits on web page’. Organisation B decides that, as the client base comprises a small number of large clients, it must focus on keeping existing clients happy. Selected *goal indicator* is ‘client satisfaction levels'. Goals for C relate to defect levels and the group decides to focus on ‘defects found during testing’.

4.3 Step 3: Choose activities to close gap

We finally work with the organisation to establish appropriate activities to close gaps between the current and desired values of *goal indicators*. To help inform choice, we consider relevant *context indicators*, existing standards such as ISO/IEC 12207 (ISO 1997) and the organisational literature.

Organisation A is structured into marketing, development and QA teams and has in place some sound development processes. We understand from the literature that “more flexible product development procedures are important to the success of new products in dynamic environments“ (Carbonell & Rodriguez-Escudero, 2009, p. 32, citing Henard & Szymanski, 2001) and that innovation effectiveness is supported by the use of cross-functional teams in conjunction with strong management support (p. 29, citing Cooper & Edgett, 2008). We suggest that such a team be set up and supported by the owner.

For organisation B, possible activities to improve ’client satisfaction levels’ may include upgrading the product, assigning a client advocate and weekly contact. We learn that a new version of the product is pending and management, now aware of the dangers of reduced satisfaction levels, chooses to assign a dedicated client advocate to each major client during the transition period.

For C, we identify *context indicators* and values as 'low process knowledge', 'culture flexible', 'no spare time' and ’motivated’. We also learn that the source of the problem is believed to be lack of clarity about the product requirements i.e. resides in the interface between product definition, development and QA. We decide that the most appropriate way to support process change is to provide options relating to the situation and work with group members to establish the most acceptable option(s). The group decides to hold a weekly meeting at which uncertainties in features will be identified and a senior member assigned to flesh out features, if deemed necessary.

5 Case study

In this Section, we describe how the model was retrospectively applied in the context of a software process improvement initiative in a small software organisation in Auckland. For reasons of confidentiality, only relevant aspects of the study are reported.

As is common for small organisations, members of the target team had an in-depth knowledge of the product and client base. Each member 'owned' one or more roles that included development with both existing and new technologies, testing and support for the client-facing sections of the organisation. Management was very happy with the group's performance and simply wanted to confirm that nothing important was being missed. Interviews aimed at understanding strategic objectives were held with the management team and individual interviews aimed at uncovering potential issues were held with group members. These were followed by two group sessions aimed at consolidating and agreeing on issues and brainstorming appropriate solutions.

The ISO/IEC 12207 model (ISO 1997) was applied in the backgound as reference model. However, the target team operated at a very immature level with respect to this model, with virtually no process areas formalised at any level. Regardless of this, the team appeared to function well within the existing setup - no-one had any complaints about quality or delivery schedules and the team was largely happy with how things were. The author involved in the initiative first attempted to understand expectations for change, as a knowledge that the status quo was about to change would hopefully help both management and team to understand the need for implementing some basic processes. Management had plans for product growth and thought the team 'might grow' but didn't expect this would affect performance. During individual team interviews, the likelihood of team growth was presented and members asked to identify issues that might occur should this happen. Thirty-one issues were identified: twenty-six relating to growth scenario, one relating to product strategy and four relating to current issues. During team brainstorming, 'solutions' were identified and included, for example, formalisation of a team space as mitigation for cultural issues on growth, strategies for inconsistent coding style and gold-plating and the introduction of more formal version control, build, defect tracking and testing processes. Brainstorming effectively addressed contextual considerations.

A simple gap analysis with standard models simply did not help as a result of the immaturity of the organisation (they 'didn't know what they didn't know'). In order to support progression, it was necessary to establish a motivation for change (*increase in size*), support the team in identifying pending issues (*goal indicators)* and help them brainstorm ways to address these.

Both team members and management appeared 'happy' with resulting recommendations and reported plans to action these. No followup has been carried out, as yet, and so the success of the initiative is not yet certain. However, it became apparent during interviews with management that the expectation was that team would continue to contribute towards product strategic direction, a *growth* situation according to our model. Athough the approach taken supported recommendations that appeared to be appropriate for the team at that point in time, our model leads us to believe that very little will have changed and the success of the initiative will have been minimal.

6 Discussion and future work

The model presented in Section 4 has been created as a result of our experiences with local New Zealand software organisations. At this stage, the model has been tested only informally. We now plan to formally test some hypotheses based on the model, as discussed below.

Our first observation relates to the ’manufacturing process’ source of the popular process improvement models, such as CMMI and ISO/IEC 12207. We suggest these models are based on an assumption of stable product development whereas many small organisations are characterised by innovation and creativity. We believe the mis-match may be a contributing factor in failed SPI initiatives. Application of our model involves first identifying *growth* (i.e. business-not-as-normal) situations. We hypothesise that small organisations characterised by *growth* are less likely to achieve successful SPI outcomes because efforts must be focussed elsewhere. Our interest in this hypothesis relates to preventing doomed SPI initiatives with corresponding loss of money, time and morale.

Our second observation concerns the need to identify which goals are most important and focus improvement efforts on meeting these. Traditional models contain an implicit assumption of cost and quality related goals and the risk is that simple solutions, such as assigning a client advocate to promote client satisfaction, will be missed. The standard reference models mandate which processes are acceptable and do not support, for example, weekly meetings to clarify requirements. The 'blueprint' approach of traditional models means that, even if the traditional models were to include all kinds of activities and key indicators, they simply do not go far enough as they do not help organisations decide which gaps to close. I do not want to improve my testing process if the problem lies in clarity of requirements or if the test team is overworked and annoyed. We hypothesise that the outcomes of SPI initiatives are more likley to be favourable if recommendations are based on the identification of *goal indicators*, root causes and *context indicators*.

The key contribution of this paper is the provision of a model from which we may create and formally test hypotheses with the aim of improving our understanding of the issues surrounding SPI initiatives.

7 Summary

We have suggested that a suitable model for a software system that will provide support for SPI initiatives is that of 'software system health'. The health of a human changes through time as changes to body, consciousness or environment occur. In an analogous way, the health of a *software system* changes through time as values of *key indicators* for any of *software product*, *software product owner* or *stakeholders* change. An SPI initiative may occur at any point in the *software system lifecycle* and must take into account the motivation for change, i.e. *sickness*, *growth* or *adaptation*, the *goal indicators* that inform a focus for change and the *context indicators* that must be taken into account when identifying what to change and how to change it. We have applied the model retrospectively to an SPI initiative in a small local software organisation. We have identified two hypotheses based on the model and plan to test these within local software organisations.

8 Literature

Chrissis MB, Konrad M, Shrum S. *CMMI Second Edition Guidelines for Process Integration and Product Improvement.* Addison-Wesley: Massachusetts USA, 2007.

International Standards Organisation (ISO). *ISO/IEC 15504.5: Information Technology – Process Assessment – An exemplar Process Assessment Model.* The International Standards Organisation, 2006.

International Standards Organisation (ISO). *ISO/IEC 12207: Information Technology – Software Lifecycle Processes.* The International Standards Organisation, 1997.

Cater-Steel A. Process Improvement in Four Small Software Companies. *Proceedings of the Australian Software Engineering Conference (ASWEC'01)* **2001,** pp. 262-272.

Grunbacher P. A software assessment process for small software enterprises. *Proceedings of the 23rd EuroMicro Conference (EUROMICRO97)* **1997**, pp. 123-128.

Huack JCR, von Wagenheim CG, de Souza RH, Thirty M. Process Reference Guides – Support for Improving Software Processes in Alignment with Reference Models and Standards. *Communications in Computer and Information Science (CCIS)* **2008; 16,** pp. 70-81.

McCaffery F, Pikkarainen M, Richardson I. AHAA – Agile, Hybrid Assessment Method for Automative, Safety Critical SMEs. *Proceedings of the 30th International Conference on Software Engineering (ICSE'08)* **2008**, pp. 551-560.

Laporte CY, Alexandre S, O'Connor RV. A Software Engineering Lifecycle Standard for Very Small Enterprises. *Communications in Computer and Information Science (CCIS)* **2008; 16,** pp. 129-141.

Chen X, Staples M, Bannerman P. Analysis of Dependencies between Specific Practices in CMMI Maturity Level 2. *Communications in Computer and Information Science (CCIS)* **2008; 16,** pp. 94-105.

Aaen I. Software Process Improvement: Blueprints versus Recipes. *IEEE Software* **2003; 20(5)**, pp. 86-93.

Kirk D. *A Flexible Software Process Model.* PhD thesis, University of Auckland: 2007.

MacDonell S, Kirk D, McLeod L. Raining Healthy Software Systems. *The 4th International ERCIM Workshop on Software Evolution and Evolvability (Evol'08)* **2008,** pp. 21-24.

Pikkarainen M, Salo O, Still J. Deploying Agile Practices in Organizations: A Case Study. *Lecture Notes in Computer Science (LNCS)* **2005; 3792**, pp. 16-27.

MacCormack A, Kemerer C, Cusumano C. Trade-offs between Productivity and Quality in Selecting Software Development Practices. *IEEE Software* **2003; 20(5)**, pp. 86-93.

Carbonell P., Rodriguez-Escudero A.I. Relationships among team's organizational context, innovation speed, and technology uncertainty: An empirical analysis. *Journal of Engineering and Technology Management* **2009; 26**, pp. 28-45.

9 Author CVs

Diana Kirk

Diana Kirk is Research Fellow at the Software Engineering Research Laboratory (SERL) at the Auckland University of Technology (AUT) in New Zealand. Diana was awarded a Masters in Computer Science from the University of Edinburgh and a Doctorate from the University of Auckland in 2007. Her industry experience spans twenty years and includes technical programming, co-directing an emerging company, project management, software quality management and consulting. Diana's main interests relate to maximising effectiveness in the processes applied during software projects.

Stephen MacDonell

Stephen MacDonell is Professor of Software Engineering and Director of the Software Engineering Research Laboratory (SERL) at the Auckland University of Technology (AUT) in New Zealand. Stephen was awarded BCom(Hons) and MCom degrees from the University of Otago and a PhD from the University of Cambridge. He undertakes research in software metrics and measurement, project planning, estimation and management, software forensics, and the application of empirical analysis methods to software engineering data sets.

SPI Based Learning Organisations

Richard Messnarz[1], Gearoid O'Suilleabhain[2], Damjan Ekert[1]
[1]ISCN GesmbH, Schieszstattgasse 4, A-8010 Graz, Austria
Tel: +43 316 811198, Fax: + 43 316 811312, Email: rmess@iscn.com
Gearoid O'Suilleabhain, Cork Institute of Technology, Ireland

Abstract

Most improvement initiatives focus on assessments and a derived improvement plan. However, the effort to really implement and sustain improvements is much bigger than the assessment effort. Also, it is crucial for SPI success that improvements are not just a collection of formal documentation requirements but show real benefit for the development and help to optimize the development.

In this paper we describe the relationship between learning organisations and SPI initiatives, and provide best practice examples of their implementation in industry.

The SPI based learning organisation applies strategies for the design and selection of best practices, knowledge management, learning portals and learning strategies, to include all working places in a collaborative SPI knowledge sharing environment.

This paper shows how these principles have been applied and how such learning organisations and underlying platforms have been created.

Keywords

Process Improvement, Learning Organisations, Integrated SPI Learning Platforms, Experiences

1 Selection and Building of Best Practices – Knowledge Components Collection

A best practice [3] [5] [7] [9] which will be accepted by all levels of staff and which is a useful core competence in a learning organisation strategy has the following features F1 to F3:

- F1: For developers the best practice has practical value to improve the existing development work, probably even reduces the effort, and can be demonstrated by development examples.
- F2: For managers the best practice helps to save time and cost and to increase quality (cost factor x) and can be multiplied in many projects, thus enabling a multiple benefit of the cost factor x [1].
- F3: For assessors the best practice helps to demonstrate improved capability.

Only if a best practice fulfils all three F1 – F3 criteria it should become a knowledge component of the SPI based learning organisation. Too many times we see industry improvement programs that claim to fulfil F1-F3 but actually only help to prove F3.

Example 1 - Code Reviews

It is relatively easy (for any consultant to recommend) to come up with a code review checklist and ask for formal code reviews to be planned in the review plan as part of the quality plan. In practise we always had difficulties to explain developers to actually invest into this effort, especially if 99% of the formally filled review checklists only confirm a simple “ok” and do not identify problems. In practise we needed to find a synergy between the development environment, the development approach and the benefit the developers would gain by reviews.

Example Experience: In one case (automotive development) the functional software was developed by a model driven approach and code generators. So we came up with a complexity metric for the models and measured the design model complexity of modules. Then we mapped the errors onto the modules and found that with the exception of the input and output related modules the correlation worked. So we decided that modules above a certain complexity value needed to be reviewed. This approach then led to early re-design of modules that caused errors. Since the developers experienced the value of this analysis in a complex and safety-critical development, the project started to plan reviews regularly.

→ Knowledge Component:

- How to measure design model complexity and how to automatically extract this from properties of models implemented in Matlab;
- threshold levels applied on complexity values;
- Review performance and review examples.

Example Experience: In another case (Microsoft framework and Internet software) the developers used semantic editors enforcing the coding guidelines (programming guideline mistakes showed as red) and semantic compilers and tracking of the trend of the semantic compiler error and warning feedback (nightly builds). The coding rules were put into rules for the semantic compiler.

→ Knowledge Component:

- Standard configuration files to be used in the IDE (Integrated Development Environment) to enforce the coding guidelines;
- how to use e.g. CheckStyle (tool to automatically do a semantic check on the syntax) and how to automatically call analysis tools to provide a report about errors

and warnings;

- how to use CruiseControl (Tool to support continuous tests and integration) and run CheckStyle with every build.

Example 2 - Requirements Management

It is relatively easy (for any consultant to recommend) to come up with what is written directly in the standard: You must document user requirements and link them with system requirements including acceptance criteria. You must link system with software requirements including acceptance criteria. And so forth. It is much tougher to develop a requirements management process and solution which helps to achieve higher performance of technical work and saves cost, and at the same time reaches a high capability level in CMMI / SPICE.

Example Experience: In one case (automotive development) a number of projects were based on the same core system and core software – a family of products belonging to the same product line. The idea was to analyse all projects and identify those 90% functions which are used across products and put together a generic re-usable system and software specification. Based on that a re-usable test scope including acceptance criteria was developed. The improvement project then delivered consistent requirements and a test scope across a family of products and saved documentation effort by 90% for future products from that family.

→ Knowledge Component:

- Re-Usable Specification across products in a product family;
- re-Usable test scope across products in a product family;
- concepts to create and support product lines in a requirements management system.

Example Experience: In a commercial multimedia software development case the customer (long standing) always provided framework contracts with no detailed content definition (budget per annum for further development). The project had constant problems in budget overruns due to the fact that the customer always stressed to include more and more features with fixed budget. In this case the requirements improvement led to a model for joint requirements workshops (constructive communication with customers) where requirements were put into a pool of features and jointly prioritized and tagged with effort. This led to a joint view of expectations and a clear identification of customer priorities (invest in features most important to satisfy the long standing customer first).

→ Knowledge Component:

- How to run requirements workshops;
- how to create a baseline requirements specification (pool of features) and how to assign them by priority methods into annual framework master releases;
- methods to prioritize requirements.

This “bricks of knowledge approach” forms a constructive step by step method to build a library of SPI knowledge with re-usable best practices in a firm [6].

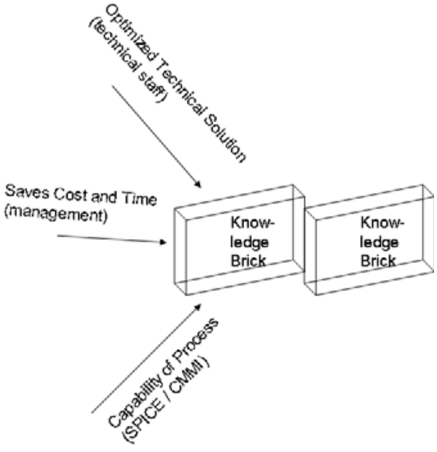


Figure 1: The knowledge bricks approach

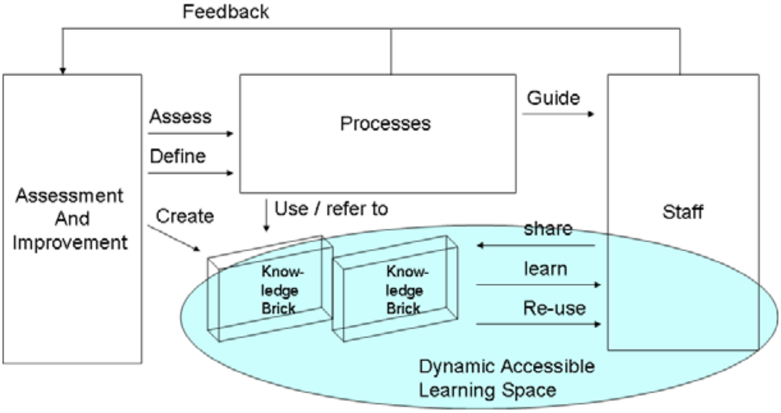


Figure 2: The processes and the knowledge bricks

Processes describe objectives, roles, activities, results, etc. Knowledge bricks describe real useful and practice oriented work patterns which help to fulfil the technical and process goals at the same time. This synergy is a major driver that “theoretical” processes get acceptance by “practical” staff.

2 Adapting Learning Organisation Models for SPI

As published in previous articles when applying a learning organisation strategy a company becomes an ORGANISM where through continuous learning spirals the knowledge grows and the core competences increase continuously.

The learning organisation [2] [4] [8] [12] [13] strategies focus on

- the selection of core knowledge where the company can concentrate their energy in areas where they excel or where no one else can operate;
- use of up-to-date IT infrastructure and communication strategies to multiply core knowledge into many projects / products;
- the creation of dynamic learning loops to networks and outside partnerships and customers to continuously build the core knowledge.

This general learning organisation architecture has been adapted to the SPI field in Fig. 3.

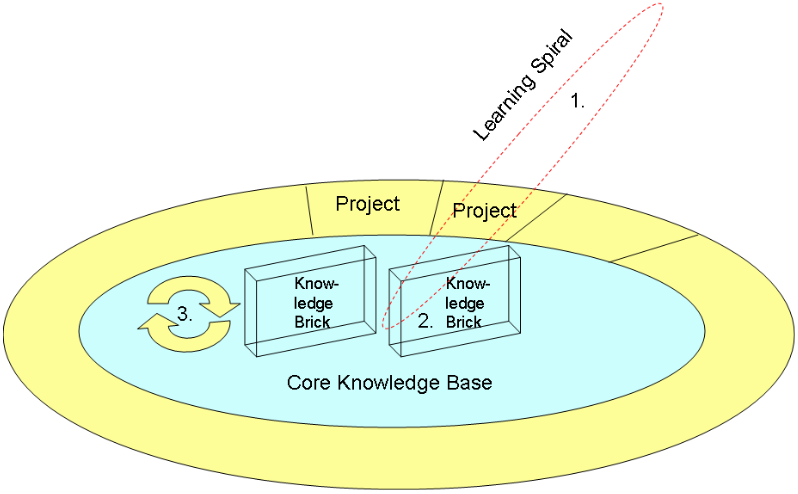


Figure 3: Integrating Learning Organisation Models with SPI Initiatives

The learning process model includes (see Figure 3):

1. A customer demands an improvement program. The project is being assessed. The project uses the existing knowledge bricks to practically implement SPICE / CMMI. The improvement plan describes a project specific set of improvement actions.
2. The SPI team in the organisation (not a project specific but a knowledge brick specific strategy!) defines a solution in form of a knowledge brick which will add value across the projects

and can be re-used [6]. The knowledge brick must satisfy the F1-F3 criteria mentioned earlier.

3. An up-to-date communication and learning infrastructure is used to multiply knowledge bricks into the projects.

This approach has a direct impact on the assessment infrastructure [9], because learning from best practices must be supported.

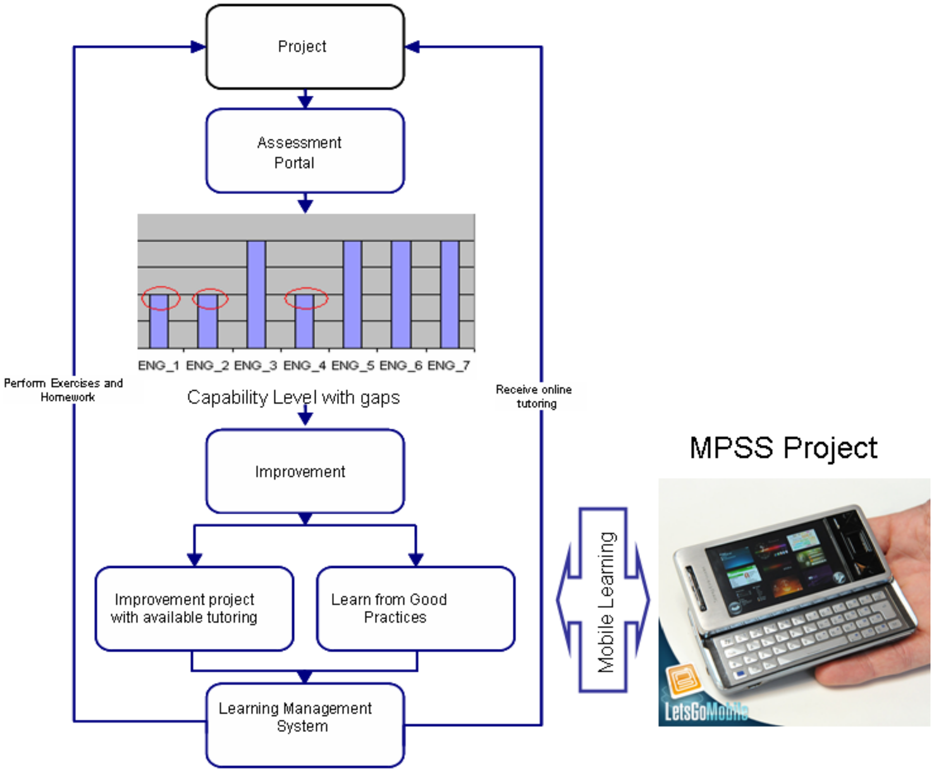

Project
Assessment Portal
ENG_1 ENG_2 ENG_3 ENG_4 ENG_5 ENG_6 ENG_7
Perform Exercises and Homework
Capability Level with gaps
Receive online tutoring
MPSS Project
Improvement
Mobile Learning
Improvement project with available tutoring
Learn from Good Practices
Learning Management System
LetsGoMobile

Figure 4: Example Structure of an Adapted Assessment and Learning Infrastructure

Explanations to Figure 4:

An assessment portal in such a system is usually a multi-user based assessment system which supports assessment teams, rating, reporting and benchmarking.

Learning management systems are usually supporting online multimedia courses, discussions, exercises and rating of homework done, and guidance for learners. Such systems are used for the good practices roll out.

In new EU research projects in the LLP (Life Long Learning) [14] area new interfaces have been developed which allow “cool” learning of best practices through mobile phones with web browser functions (generations from 2007 onwards). Examples are Ericsson Xperia X1, Nokia E71, etc.

3 Implementation Experiences

Capability Adviser is a multi-user and SQL based assessment portal system used in middle sized and large companies for administering and performing assessments across the divisions. Assessment teams share data and reports are generated, as well as improvement tracking is supported for improvement actions expected from projects.

The learning management issue was solved by developing an interface module which allows integrating the most widely used e-learning systems with the Capability Adviser. This includes Moodle (www.moodle.com), DIPSEIL, etc.

To support such a learning organisation an infrastructure for learning has been set up and learning systems offering best practices have been installed. Experiences were collected from different learning based projects from autumn 2006 onwards.

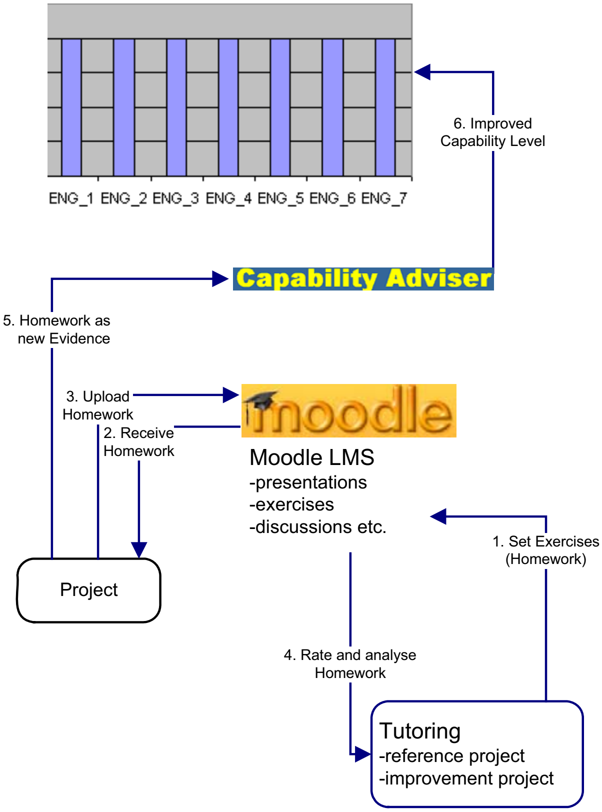
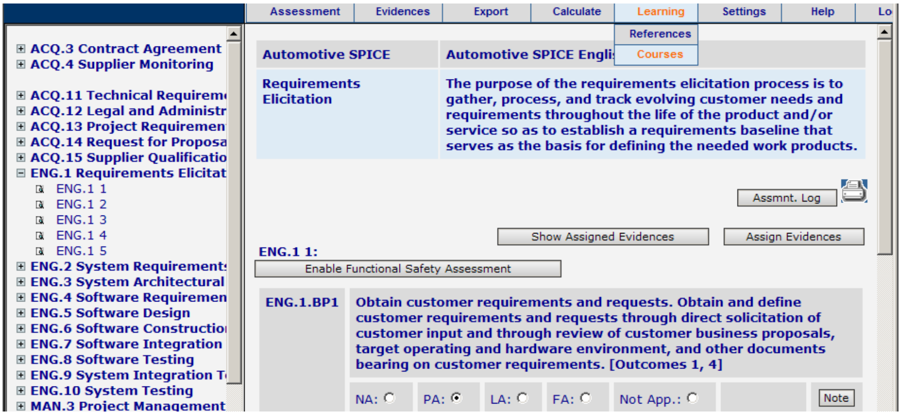


Figure 5: Overview of Learning Infrastructure Used

Multi-user based assessment portals were used. Selected members from projects (accounts to be provided) can log in and directly see the assessor recommendations. Project managers can log in and do self assessments. And a learning section was added which directly allows employees to access the learning areas.

The learning areas supported:

- From each workplace in the company people could attend 30-45 minutes online multimedia presentations including slides, sound, graphics, and animation. E.g. the course prepared by the process owner for requirements management could be attended by everyone who had to use this approach in the project. It was important that the course would be an automatic self-running multimedia show so that the presence of the requirements management process owner was not necessary.
- Attendees of the course got a homework task to apply this in the own project. They uploaded the homework to the learning system.
- The process owner and experts (configured as tutors) had agreed times (every two weeks) to discuss online 1 hour with the training attendees and give feedback to the homework.
- The system also allowed giving a grading to homework done. The grading described how well the principles were implemented.



⊞ ACQ.3 Contract Agreement
⊞ ACQ.4 Supplier Monitoring
⊞ ACQ.11 Technical Requirem
⊞ ACQ.12 Legal and Administr
⊞ ACQ.13 Project Requiremen
⊞ ACQ.14 Request for Proposa
⊞ ACQ.15 Supplier Qualificatio
⊟ ENG.1 Requirements Elicitat
ENG.1 1
ENG.1 2
ENG.1 3
ENG.1 4
ENG.1 5
⊞ ENG.2 System Requirements
⊞ ENG.3 System Architectural
⊞ ENG.4 Software Requiremen
⊞ ENG.5 Software Design
⊞ ENG.6 Software Constructio
⊞ ENG.7 Software Integration
⊞ ENG.8 Software Testing
⊞ ENG.9 System Integration T
⊞ ENG.10 System Testing
⊞ MAN.3 Project Management

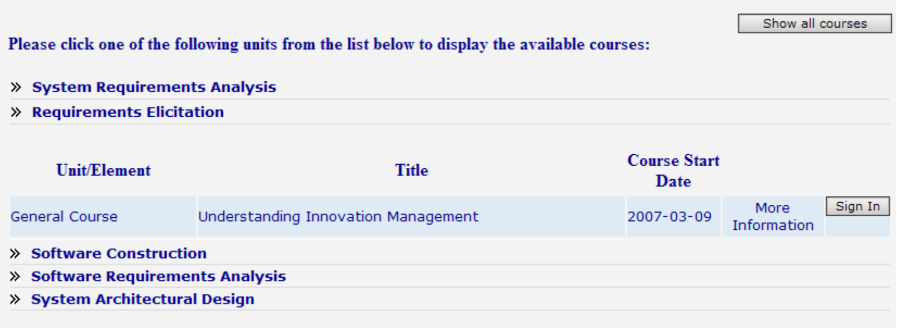
Automotive SPICE	Automotive SPICE Engli
Requirements Elicitation	The purpose of the requirements elicitation process is to gather, process, and track evolving customer needs and requirements throughout the life of the product and/or service so as to establish a requirements baseline that serves as the basis for defining the needed work products.

ENG.1 1:

ENG.1.BP1 Obtain customer requirements and requests. Obtain and define customer requirements and requests through direct solicitation of customer input and through review of customer business proposals, target operating and hardware environment, and other documents bearing on customer requirements. [Outcomes 1, 4]

NA: ○ PA: ● LA: ○ FA: ○ Not App.: ○

Figure 6: Access to Courses from Processes in the (Self-)Assessment Environment



Please click one of the following units from the list below to display the available courses:

» System Requirements Analysis

» Requirements Elicitation

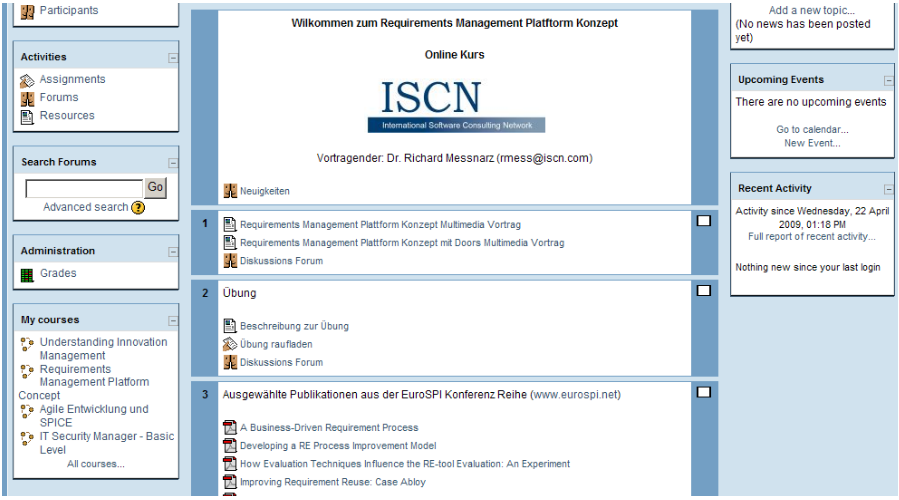
Unit/Element	Title	Course Start Date		
General Course	Understanding Innovation Management	2007-03-09	More Information	Sign In

» Software Construction

» Software Requirements Analysis

» System Architectural Design

Figure 7: Automatic Sign In into Multimedia Course Environment



Participants

Activities

Assignments
Forums
Resources

Search Forums

Go
Advanced search

Administration

Grades

My courses

Understanding Innovation Management
Requirements Management Platform Concept
Agile Entwicklung und SPICE
IT Security Manager - Basic Level
All courses...

Wilkommen zum Requirements Management Plattform Konzept

Online Kurs

ISCN
International Software Consulting Network

Vortragender: Dr. Richard Messnarz (rmess@iscn.com)

Neuigkeiten

1
Requirements Management Plattform Konzept Multimedia Vortrag
Requirements Management Plattform Konzept mit Doors Multimedia Vortrag
Diskussions Forum

2 Übung
Beschreibung zur Übung
Übung raufladen
Diskussions Forum

3 Ausgewählte Publikationen aus der EuroSPI Konferenz Reihe (www.eurospi.net)
A Business-Driven Requirement Process
Developing a RE Process Improvement Model
How Evaluation Techniques Influence the RE-tool Evaluation: An Experiment
Improving Requirement Reuse: Case Abloy

Add a new topic...
(No news has been posted yet)

Upcoming Events

There are no upcoming events

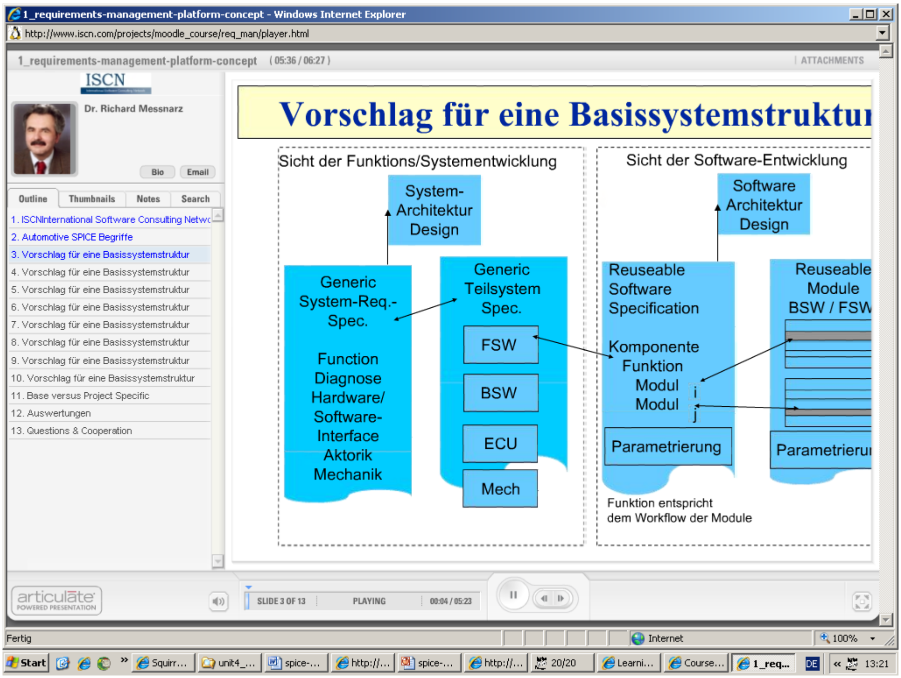
Go to calendar...
New Event...

Recent Activity

Activity since Wednesday, 22 April 2009, 01:18 PM
Full report of recent activity...

Nothing new since your last login

Figure 8: Course Environment – Example : Requirements Management Platform / Re-Use Concept



1_requirements-management-platform-concept - Windows Internet Explorer

http://www.iscn.com/projects/moodle_course/req_man/player.html

1_requirements-management-platform-concept (05:36 / 06:27) ATTACHMENTS

ISCN
Dr. Richard Messnarz
Bio Email

Outline Thumbnails Notes Search

1. ISCNInternational Software Consulting Netwo
2. Automotive SPICE Begriffe
3. Vorschlag für eine Basissystemstruktur
4. Vorschlag für eine Basissystemstruktur
5. Vorschlag für eine Basissystemstruktur
6. Vorschlag für eine Basissystemstruktur
7. Vorschlag für eine Basissystemstruktur
8. Vorschlag für eine Basissystemstruktur
9. Vorschlag für eine Basissystemstruktur
10. Vorschlag für eine Basissystemstruktur
11. Base versus Project Specific
12. Auswertungen
13. Questions & Cooperation

Vorschlag für eine Basissystemstruktur

Sicht der Funktions/Systementwicklung
System-Architektur Design
Generic System-Req.-Spec.
Function Diagnose Hardware/ Software-Interface Aktorik Mechanik
Generic Teilsystem Spec.
FSW
BSW
ECU
Mech
Sicht der Software-Entwicklung
Software Architektur Design
Reuseable Software Specification
Komponente Funktion Modul Modul
i
j
Parametrierung
Reuseable Module BSW / FSW
Parametrieru
Funktion entspricht dem Workflow der Module

Figure 9: Example – Online Lecture with Sound

Saving of Cost and Time Related Feedback

- Usually the expert or process owner is a bottleneck when it comes to roll out a method or best practice to a number of projects. By using this infrastructure based learning approach it was possible to disseminate the knowledge to all projects in a format which makes it attractive to projects. Practical experience showed that the following additional steps are necessary to ensure that projects will really use it:
    - Each project at its start must undergo a training of the best practice knowledge bricks using the infrastructure and guidance.
    - This kick off training is promoted and requested by management and senior staff.

Technical Integration Related Feedback and Acceptance by Technical Staff

- This infrastructure based learning approach allowed “work based learning” where people from each work place can participate. Developers use the environment from their PC, like they use an IDE (Integrated Development Environment). For technical people learning must be attractive, “cool”, and offer a personal development opportunity. It [14] works best if employees using the learning facility can gather “competence points” (similar to certificates in courses) which will help in their personal development inside the company (recognition by management).

Core Knowledge Related Feedback

- The whole system worked well only if the core knowledge bricks fulfilled all F1-F3 criteria. If the core knowledge is wrongly selected and prepared, the systematic learning approach will not work.
- For the selection a win (manager) – win (technical staff) – win (process experts) situation is important as we have illustrated in the first chapter of the paper.

4 Conclusions & Outlook

In 2006 – 2008 the learning systems approach has been field tested and in 2009 the systematic approach is rolled out to a group of SPI committed organisations.

However, from the experiences we have learned that in 2009 – 2010 our research should focus on areas where we can facilitate the process of selecting and preparing core SPI knowledge components.

- Currently an SPI manager qualification (with a European Certificate issued by ECQA – www.ecqa.org) is developed which will include such strategies [10].
- Currently an integrated designer qualification (with a European Certificate issued by ECQA – www.ecqa.org) is developed which will strengthen the idea of collaborative design (product – system – software) and deciding for improvements which fit both, product improvements and process goals [10].
- Currently an e-learning manager qualification (with a European Certificate issued by ECQA – www.ecqa.org) is developed which will include competencies to establish learning infrastructures [10].
- Currently we also run a mobile learning research project, where learning through mobile phones will become possible for SPICE and SPI topics (make the learning a “cool” thing for developers).
- Designing a methodological guidance for implementing core knowledge strategies in SPI and building on a collection of examples [6] [11].

5 Literature

[1] Messnarz R., Tully C. (eds.), The PICO - Book: Better Software Practice for Business Benefit - Principles and Experience, IEEE Computer Society Press, in publication

[2] A Learning Organisation Approach for Process Improvement in the Service Sector , R. Messnarz. C. Stöckler, G. Velasco, G. O'Suilleabhain, A Learning Organisation Approach for Process Improvement in the Service Sector, in: Proceedings of the EuroSPI 1999 Conference, 25-27 October 1999, Pori, Finland

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

[4] O'Keeffe, T., & D. Harrington, 2001. Learning to Learn: An Examination of Organisational Learning in Selected Irish Multinationals. Journal of European Industrial Training, MCB University Press, Vol. 25: Number 2/3/4

[5] Richard MESSNARZ, Romana Vajde Horvat, Katja Harej, Eva Feuer, Process Improvement is Not Enough – Do not Forget Innovation Processes, in: EuroSPI 2004 Proceedings, 10-12 October 2004, ISBN: ISSN-NO: 1503-416X, October 2004

[6] Hege Dreiem, Rune Myrdal, Kjetil Jørgensen-Dahl, Building a learning organisation: experience with establishing communities of practice, in: EuroSPI 2004 Proceedings, 10-12 October 2004, ISBN: ISSN-NO: 1503-416X, October 2004

[7] Matti Haavikko, Mika Johansson, Risto Nevalainen, Prioritizing SPI actions based on business needs and impacts, in: EuroSPI 2004 Proceedings, 10-12 October 2004, ISBN: ISSN-NO: 1503-416X, October 2004

[8] Richard Messnarz, Gearoid O'Suilleabhain, Ray Coughlan, From process improvement to learning organisations (p 287-294), in: Wiley Interscience Journal, SPIP Software Process Improvement in Practice Series, Volume 11 Issue 3 , Pages 213 - 335 (May/June 2006), 2006

[9] Richard Messnarz, Damjan Ekert, Assessment-based learning systems - learning from best projects (p 569-577), in: Wiley Interscience Journal, SPIP Software Process Improvement in Practice Series, Volume 12 Issue 6 , Pages 505 - 610 (November/December 2007), 2007

[10] Richard Messnarz, Damjan Ekert, Michael Reiner, Gearoid O'Suilleabhain, Human resources based improvement strategies - the learning factor (p 355-362), in: Wiley Interscience Journal, SPIP Software Process Improvement in Practice Series, Volume 13 Issue 4 , Pages 297 - 382 (July/August 2008), Jul 2008

[11] Gunther Spork, Uwe Pichler, Establishment of a performance driven improvement programme (p 371-382), in: Wiley Interscience Journal, SPIP Software Process Improvement in Practice Series, Volume 13 Issue 4 , Pages 297 - 382 (July/August 2008), Jul 2008

[12] Riel A., Tichkiewitch S., Messnarz R., Integrated Engineering Skills: Improving your System Competence Level. In: EuroSPI 2008 Industrial Proceedings, Dublin, Delta Series about Process Improvement, ISBN 978-87-7398-150-4pp. 8.9-8.20.

[13] Messnarz R ., Spork G., Riel A., Tichkiewitch S., Dynamic Learning Organisations Supporting Knowledge Creation for Competitive and Integrated Product Design. In: Proceedings of the International CIRP Design Conference, Cranfield, March 2009

[14] MPSS EU Project, 2009-2010, Mobile Performance Support System

6 Author CVs

Dr Richard Messnarz

Dr. Richard Messnarz (rmess@iscn.com) is the Executive Director of ISCN LTD. He studied at the University of Technology Graz and he worked as a researcher and lecturer at this University from 1991 - 1996. In 2 European mobility projects (1993 and 1994) he was involved in the foundation of ISCN, and he became the director of ISCN in 1997. He is/has been the technical director of many European projects:

- PICO - Process Improvement Combined Approach 1995 - 1998,
- Bestregit - Best RegionalTechnology Transfer, 1996 - 1999,
- TEAMWORK - Strategic Eworking Platform Development and Trial, 2001-2002,
- MediaISF - Eworking of media organisation for strategic collaboration on EU integration, 2001-2002

He is the editor of a book "Better Software Practice for Business Benefit", which has been published by IEEE (www.ieee.org) in 1999 (the leading research publisher in the USA). He is the chairman of the EuroSPI initiative and chair of the programme committee of the EuroSPI conference series.

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

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

Dipl.Ing. Damjan Ekert

Dipl.-Ing. Damjan Ekert studied Telematics at the University of Technology, Graz, Austria. He finished his studies with distinction. Since 2001 he works for ISCN as SW project manager and SW integrator. He is a certified e-Security Manager and European Project Manager Trainer. For the last three year he has been working as a consultant in the field of Software Process Improvement, he is member of the S2QI Initiative, certified ISO15504 Assessor, member of the European Certification and Qualification Association and vice- president of the ISECMA e.V. (International IT Security Management Board).

The SPI Professional Qualification Scheme

Tomas Schweigert, SQS, Germany
Richard Messnarz, ISCN, Austria + Ireland
Morten Korsaa, DELTA, Denmark
Jorn Johansen, DELTA, Denmark
Risto Nevalainen, STTF/SPINet, Finland
Miklos Biro, Corvinus University, Hungary

EU Certificates Contact for SPI Manager (www.ecqa.org):
Tomas Schweigert
SQS Software Quality Systems AG
Stollwerckstraße 11
51149 Köln
tomas.schweigert@sqs.de

Abstract

While there is a wide spread knowledge about the performance of assessments the real implementation of SPI (Software, System and Service Improvement) takes a lot of effort and experience to achieve an ROI (Return on Investment). Assessor qualification is currently in the focus. However, to ensure the success of the improvement actions another additional qualification is needed, which is the (SPI) Software, System and Service Improvement Manager. The paper explains the goals and procedures of the EU Certification and Qualification Association (ECQA). It describes the EU qualification scheme for Software, System and Service process improvement professionals and the need for the SPI qualification. The paper also gives a short overview over the topics of the SPI qualification and the included job roles.

Keywords
SPI; Job Role Committee; Skill Card; European Qualification

1 Introduction

70% of all SPI initiatives fail [Statz]. The more formal reasons are reported as lack of management commitment or unrealistic expectations [Statz]. The reasons behind this is an invalid understanding of the job roles in the SPI context and poor role training and qualification schemes for SPI professionals. There are qualification schemes available for some life-cycle models like V-Modell XT, and there are qualification schemes for software process assessments like ESI, SEI, INTACS or IntRSA in place. But there is no scheme available for the approx. 10.000 people who work on the improvement of software system or service (S3) processes in the whole of Europe. Therefore, there is a need for a professional job role scheme. To make it valid for the holders of the certificate as well as the employing organisations and to stay in line with the development of ISO/IEC 15504, the qualification addresses the software, the system and the service process area.

2 The EU Certification and Qualification Association

2.1 Europe Wide Certification

- The ECQA is the result of a number of EU supported initiatives in the last ten years where in the European Union Life Long Learning Programme different educational developments decided to follow a joint process for the certification of persons in the industry.
- Through the ECQA it becomes possible that you attend courses for a specific profession in e.g. Germany, Hungary, Poland, etc. and perform a Europe wide agreed test at the end of the course. The certificate will then be recognized by European training organizations and institutions in 18 member countries.

2.2 Why is such a Certificate of Interest?

European work forces are highly flexible and need to work for industries across Europe (Germany, France, …). Imagine that you are attending an SPI manager course at a certified training provider in e.g. Germany and that you perform and pass the test at the end of the course. The certificate will then be recognized by training institutions and certification bodies and networks in 18 European countries. This will automatically lead to a higher recognition of the certificate and higher chances of working for customers in an open European market.

2.3 Access to a Vast Pool of Knowledge

ECQA currently supports 16 professions in Europe and with the continuous support until 2012 by the European Commission the pool is growing to 21 certified professions in Europe. ECQA offers certification for professions like IT Security Manager, Innovation Manager, EU project manager, E-security Manager, E-Business Manager, E-Strategy Manager, SW Architect, SW Project Manager, IT Consultant for COTS selection, Internal Financial Control Assessor (COSO/COBIT based), Interpersonal Skills, Scope Manager (Estimation Processes), Configuration Manager, and SPI Manager. Currently new professions such as Integrated Mechatronics Designer, E-Learning Manager and Terminology Manager are being integrated until 2010.

2.4 ECQA Principles

- Skills Sets: A defined set of quality criteria has to be followed to create the learning objectives and syllabus for new professions. Only skills sets which fulfil the defined criteria are accepted by the ECQA.

- Job Role Committees: European consortia are built per accepted professions to annually update the sills set and create a European wide test questions pool.
- European Test Pool: Assuming that a group of training bodies agreed the same skills set then students must be able to pass a test, independently from the region or country in a Europe wide scope. This is the reason why (supported by the former EQN project 2005 - 2007) a Europe wide pool of test questions for the developed skills sets plus European test portals which computer automate this test scheme have been set up and allow a cross-European Internet based collaboration.
- Learning Environment: In the EU Cert Campus Project (2008 – 2010) the existing skills and exam portals are extended to an online campus for training of trainers and the multiplication of that approach into more training bodies and more regions in Europe.
- Defined Certification Rules and Procedures: The acceptance of professions and skills sets and the certification of students is based on defined quality rules and certification procedures.
- Download the ECQA Guide

  http://www.iscn.com/projects/eu_cert/documents/ecqa-guide-version3-pdf.zip

2.5 ECQA and SPI Manager

One of the most actual professions and certificates offered by ECQA is the SPI Manager. The Job Role Committee comprises organisations which implement SPI in industry since more than 16 years and are coordinators of SPI networks.

A library of SPI knowledge more than 400 peer reviewed reports have been gathered in an experience pool (see www.eu-certificates.org, section experience pool).

SPI is being launched in Europe in 2009 in the following countries: Germany, Austria, Ireland, Denmark, Hungary, Finland, and Norway. From 2010 the ECQA network will support its broad dissemination to all 18 involved member states.

3 Introduction to the SPI Qualification Scheme

Starting in 2007 a group of leading European SPI suppliers formed an ECQA Job role committee to standardize the job roles and trainings of SPI professionals starting with the SPI manager qualification. At the beginning of 2009 the current members of the job role committee agreed on a set of topics that will form the basic knowledge of the skill set. Looking at the basic ideas of ECQA, it is not enough to train people to use one management cycle like PDCA or a process modelling tool like ARIS. To support the recognition of this profession a holistic approach is needed that covers all aspects of SPI and addresses the main cause of SPI failure. The qualification scheme also reflects that process improvers work in different roles and are focussed on different topics of process improvement.

4 Needs for real SPI Management

When we look at source like the chaos report we have a stable set of up to 30 % of projects that fail and another 30% with delay, poor product quality and budget overrun. The typical reaction of organisations:

- Invest in new technologies
- Formalize processes

In most cases this approach ends up in a mail stating that the projects have to use a new tool and that new and binding processes are published at the intranet.

At the end the projects realise –if they are able to find the new processes in the intranet- that the new tool provides poor support to the new processes and has insufficient interfaces to the rest of the software development environment.

As a result of this the new tool and process is ignored or additional time and budget is spent by the project to make tool and process suitable for the needs of the project. The effect is delay and budget overrun. At the very end process performers and management loose confidence in process improvement.

To change this situation a sufficient improvement in the education of SPI managers is needed to make sure that the money spent by the management and the support of the process performers does lead to success and not to frustration.

5 Challenges of SPI Management

If you want to successful improve processes you have to fight several gaps and weaknesses:

- The improvement project is sold to the management with unrealistic promises or the management introduced the project setting up unrealistic expectations
- The management is not committed to an SPI initiative
- There is no proper KPI system in place that helps to understand the actual and the intended status
- There is no process data available to evaluate the results of an SPI initiative
- SPI performers are not properly trained
- Change resistance at management and performer level
- Lack of ability to deploy processes
- Poor Tailoring
- Poor Training
- Poor Audit Assessment and Feedback
- Process performers are not properly trained
- Roles and responsibilities are not clearly defined
- Training doesn’t fit to the role.
- Poor support for teamwork
- Poor marketing of SPI initiatives
- Lack of understanding other cultures
- Poor conflict management.

All those challenges require a well trained SPI team in which the SPI Manager has an important role. To set up an adequate training the skill set has to give an answer to the predescribed common challenges of SPI initiatives.

6 The SPI Knowledge areas

The SPI profession is built up from a wide range of knowledge topics:

- Key success factors of SPI
- Satisfiers and dissatisfiers of SPI
- Knowledge management
- Cultural aspects of SPI management
- Conflict management / mediation
- Organisational change management
- Motivation

- Teamwork
- Innovation and SPI
- Process, assessment & life cycle models
- SPI strategies
- Process modelling
- Process management
- Process deployment
- Typical techniques used in the SPI context
  - Process description techniques
  - Process improvement techniques

These topics are structured in a scheme of units and learning elements

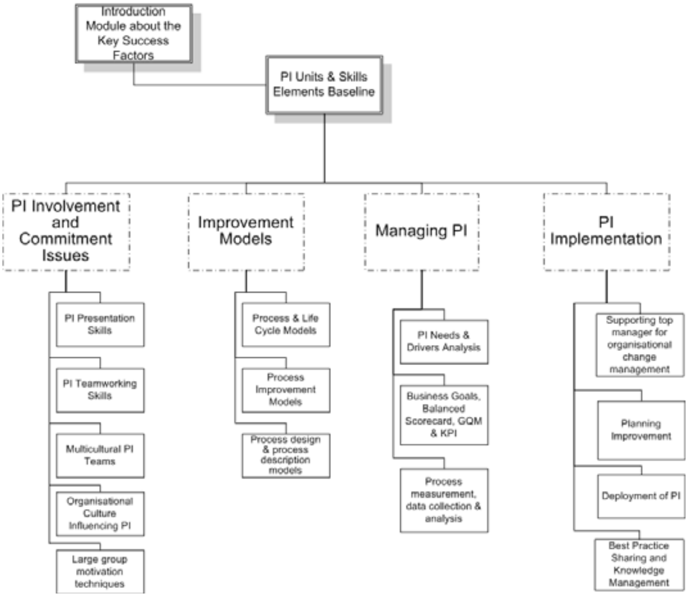

Introduction Module about the Key Success Factors
PI Units & Skills Elements Baseline
PI Involvement and Commitment Issues
Improvement Models
Managing PI
PI Implementation
PI Presentation Skills
PI Teamworking Skills
Multicultural PI Teams
Organisational Culture Influencing PI
Large group motivation techniques
Process & Life Cycle Models
Process Improvement Models
Process design & process description models
PI Needs & Drivers Analysis
Business Goals, Balanced Scorecard, GQM & KPI
Process measurement, data collection & analysis
Supporting top manager for organisational change management
Planning Improvement
Deployment of PI
Best Practice Sharing and Knowledge Management

Fig. 1. The Structure of the SPI Manager Qualification and Certification

The skill set is designed to support the SPI Manger qualification as a whole. So, instead of additional topics for advanced and expert skills, deeper understanding and proven practical experiences are required in line with the Taxonomy of Educational Objectives, often called Bloom's Taxonomy whose cognitive domain identifies six levels: knowledge, comprehension, application, analysis, synthesis, and evaluation

[http://en.wikipedia.org/wiki/Taxonomy_of_Educational_Objectives].

There is also a set of prerequisite knowledge defined (e.g. standard project or line management)

The SPI Manager Qualification Scheme

The SPI Professional qualification scheme reflects that there are several types of SPI organisation and role names used. The Job Role Committee decided not to develop a generic set of role types but to focus on the SPI Manager and support typical roles in the business field:

The SPI Professional qualification scheme will support the following job roles in the SPI context:

- SPI Manager
  (This qualification might be provided at a basic, advanced or expert level)
- SPI Coach
- Process Owner
- Process Expert
- Process Improvement Team Member
- Pilot Process Team Member

The scheme is open to integrate other roles identified as necessary for successful process improvement and implementation.

The Job Role Committee also decided not to define or support roles that are already available at the market (e.g. Assessor)

Currently the focus is to implement the SPI Manager qualification scheme. Development of other levels will be future work. Distinguishing between theses levels will be cleary guided by blooms taxonomy.

7 European Experience Pool

The partnership developing the profession has created in the last 16 years one of the largest experience pools in Europe.

Three European knowledge portals (SPIRAL - French community, EuroSPI - European Systems and Software Process Improvement and Innovation Initiative since 1994, and GNOSIS - a knowledge library of the Scandinavian community) have been configured to allow access to key knowledge underlying the offered professions. At the moment approx. 500 experience reports from industry and academia can be accessed via this. Also for about 300 European experience reports summaries are available in French, Spanish, in addition to the English version.

EuroSPI

- Access to the EuroSPI Library (http://library.eurospi.net), BROWSE by Topics, and username: organic2006, password:xxyyzz123

Gnosis

- Access to the GNOSIS Library (http://www.gnosis.fi/xitem/searchResults.do) , LOGIN with username: eqnuser, password: eqn2007)

SPIRAL

- SPIRAL RSS Flow Library (http://www.cassis.lu/rssfeed.nsf/rss?openagent&uid=06E87E56B8C1FEC6C1257361002D37B9)

These 3 libraries joined a strategic collaboration in EQN (2005 - 2007) and will continue that successful bond in the EU Cert Campus project (2008 - 2010). Annually approx. 100 experience reports and knowledge items are added to the libraries. By the end of EU Cert Campus it is expected to have achieved approx. 800 experience reports and knowledge items for the EU industrial community.

From that experience pool case studies and best practices for the SPI implementation are used for the educational programme.

8 Summary

The SPI qualification is a multi level qualification that addresses typical roles and skill sets in the SPI business. It is supported by the ECQA in order to standardise the qualification and make it available to the European labour market. In the basic level, the complete content is delivered. For the advanced levels, deeper learning and practical experiences are required.

9 Bibliography (references and further reading)

Abrahamsson, P.; Baddoo, N.; Margaria, T.; Messnarz, R. (Eds.), Software Process Improvement - 14th European Conference, EuroSPI 2007, Potsdam, Germany, September 26-28, 2007, Proceedings, Springer Verlag, Series: Lecture Notes in Computer Science , Subseries: Programming and Software Engineering , Vol. 4764

Anne Mette Jonassen Hass, Requirement Development and Management Datateknisk Forum rapport DF-17

Anne Mette Jonassen Hass, Configuration Management: Principles and Practice Addison-Wesley, 2003

Biro Miklos, Messnarz Richard (eds.), Wiley SPIP (Software Process Improvement in Practice) Series, Volume 13 Issue 6 , Pages 473 - 544 (November/December 2008) Special Issue on Industrial Experiences in SPI, 2008

Biro Miklos, Messnarz Richard (eds.), Wiley SPIP (Software Process Improvement in Practice) Series, Volume 12 Issue 6 , Pages 505 - 610 (November/December 2007) Special Issue on Industrial Experiences in SPI, 2007

Biro Miklos, Messnarz Richard (eds.), Wiley SPIP (Software Process Improvement in Practice) Series, Volume 11, Issue 3 (June 2006) , Industry Proceedings - Special Issue on Industrial Experiences in SPI, 2006.

EuroSPI Online Library, EuroSPI – European Systems and Software Process Improvement Initiative, http://library.eurospi.net, User: spicebook, Password: contilevel3

Bibliothek mit über 400 Erfahrungsberichten von Firmen und Forschungsinstituten zu Prozessverbesserungsmethoden und deren Umsetzung.

Rory O'Connor, Nathan Baddoo, Kari Smolander, and Richard Messnarz, (Eds.), Software Process Improvement - 15th European Conference, EuroSPI 2007, Dublin, Ireland, September 3-5, 2007, Proceedings, Springer Verlag, Series: Communications in Computer and Information Science , Vol. CCIS16

Richardson Ita, Runeson Per, Messnarz Richard, (Eds.), Software Process Improvement - 13th European Conference, EuroSPI 2006, Joensuu, Finland, October 11-13, 2006, Proceedings, Springer Verlag, Series: Lecture Notes in Computer Science , Subseries: Programming and Software Engineering , Vol. 4257

Ita Richardson, Pekka Abrahamsson, and Richard Messnarz (Eds.), Software Process Improvement - 12th European Conference, EuroSPI 2005, Budapest, Hungary, November 9-11, 2005, Proceedings, Springer Verlag, Series: Lecture Notes in Computer Science , Subseries: Programming and Software Engineering , Vol. 3792

Jan Pries-Heje & Jørn Johansen, ImproveIT: a book for improving software projects DELTA, 2007, IISBN 978-87-7398-086-6

Lars Mathiassen et. al., Improving Software Organizations: From Principles to Practice, Addison-Wesley, 2002, ISBN 0-201-75820-2

Laporte, C.Y., Renault, A., Desharnais, J. M., Habra, N., Abou El Fattah, M., Bamba, J. C., Initiation Software Process Improvement in Small Enterprises: Experiment with CETIC's Micro-Evaluation Framework, SWDC-REK, International Conference on Software Development, University of Iceland, Reykjavik, Islande, 27 May – 1 June 2005.

Pasi Kuvaja, Jorma Palo, Adriana Bicego, TAPISTRY—A Software Process Improvement Approach Tailored for Small Enterprises, Software Quality Journal Verlag Springer Netherlands ISSN 0963-9314 (Print) 1573-1367 (Online) Heft Volume 8, Number 2 / Oktober 1999 DOI 10.1023/A:1008909011736

Seiten 149-156

Philipp Michael, Schweigert, Tomas SPI Consulting in a Level 1 Company (An Experience Report) – PROFES Riga 2008

Scott Hanson, Alec Dorling, Tomas Schweigert, James Lovett, Assist – Leitfaden für die Beschaffung von IT für mittelständische Unternehmen, Amersham 1999

Schweigert Tomas, SPI from the Customers View - Experiences from an EUR 6 Mio Tender – EURO SPI Potsdam 2007

Schweigert Tomas, SPI – A Measurement driven Approach – SPICE DAYS Frankfurt (M) 2007

Schweigert Tomas, SPICE Support for Fix Price Projects – ICCSEA Paris 2007

Statz Joice, Oxley Don, O'Toole Patrick, Identifying and Managing Risks for Software Process Improvement, 1997

THE STANDISH GROUP REPORT CHAOS 1995 -2006

Tim Kasse, Practical insight into CMMI: The look and feel of a successful implementation, Artech House Publishers, 2004, ISBN 1-58053-625-5

Tom Gilb: Project Failure: Some Causes and Cures By Tom Gilb Web publishing 2004.

William A. Florac et. al., Measuring the Software Process: Statistical Process Control for Software Process Improvement, Addison-Wesley, 1999, ISBN 0-201-60444-2

10 Major Causes of Project Failure Centreline Solutions Inc. Web publishing 2004

10 Authors' biographies

Tomas Schweigert

Current Company:	SQS Software Quality Systems AG
Position:	Principal Consultant
Education:	Study of Law at University of Cologne
Degree:	2nd legal Degree Law (Exam)
Other Certificates:	PMP SPICE Principal Assessor
Work Experience:	Lawyer Software Quality Consultant BOOTSTRAP Lead and SPICE Principal Assessor

Dr Richard Messnarz

Current Company:	ISCN www.iscn.com

Position: Director

Education: Dr Technical Science – Informatics and Mathematics, University of Technology Graz

Degree: Dr Techn. Dipl. Ing.

Other Certificates:
INTACS Certified ISO 15504 Principal Assessor
INTACS Certified Principal Automotive SPICE ® Assessor
ECQA Certified Innovation Manager
ECQA Certified EU Project Manager
ECQA Certified e-Security Manager

Work Experience:
1990 – 1996 Researcher and Lecturer at TU Graz
1997 – now Director of ISCN
Prime Manager of ORGANIC – EU project to develop the innovation manager (2003 – 2006)
Technical Coordinator of EQN – European Quality Network (2006 – 2007)
Technical Manager of EU Certificates project With 22 EU partners (2008 – 2010)
Program Director of EuroSPI since 1994 (European Systems and Software Process Improvement, www.eurospi.net)
Vice President of ECQA (European Certification And Qualification Association) since 2007
Editor and co-writer of various books: example http://www.dpunkt.de/buecher/3069.html

Risto Nevalainen

Current Company and position: Finnish Software Measurement Association FiSMA ry, Senior Adviser and Spinet Oy, Partner

Education: Lich.Tec., Helsinki University of Technology

Other Certificates: ISO9001 Lead Auditor, Competent SPICE Assessor

Work Experience:
Assessor and Consultant in ISO9001, SPICE and CMMI related topics since 1990
Information Society activities and projects in Finland since 1979

Methodology for Process Improvement through Basic Components and Focusing on the Resistance to Change

Calvo-Manzano Jose A.[1], Cuevas Gonzalo[1], Gómez Gerzon[2], Mejia Jezreel[1], Muñoz Mirna[1], San Feliu Tomás[1], Sánchez Angel[3]
[1]Faculty of Computer Science, Polytechnic University of Madrid
{joseantonio.calvomanzano, gonzalo.cuevas, ma.munoz, tomas.sanfeliu}@upm.es,
jmejia@mpsei.fi.upm.es
[2]Universidad Autónoma de Tamaulipas, Unidad Reynosa Rodhe
ggomez@uat.edu.mx
[3]everis Foundation,
Angel.Sanchez@everis.com

Abstract

This paper describes a methodology that implements a smooth and continuous process improvement, depending on the organization's business goals, but allowing users to establish their improvement implementation pace. The methodology focuses on a process improvement basic component known as "best practice". Besides, it covers the topics: knowledge management, change management and multi-model environment. It includes both the methodology description and the results of a case study on project management process improvement in an organization.

Keywords

project management; best practices; knowledge management; change management; process improvement; multi-model environment

1 Introduction

For many organizations, the lack of good project management is the main cause of projects failure; a similar situation is found when comparing projects performance: the projects can rarely be completed in time and within an acceptable cost [1, 2].

Regarding this, authors such Pressman [3], Walter [4] and Yager [5], argue that after 30 years of having identified “the software crisis”, the problems related to project failures still continue. Moreover, Williams [6] mentions that nowadays the organization’s management has become more project-based. As a result, the need to manage projects successfully in the organizations is constantly increasing. However, if most of the organizations continue to build software products in the same way as they did for years without analyzing whether their management processes are adequate or obsolete, how do we satisfy this need? [7].

Even there is a growing group of process improvement success stories [8, 9], introducing processes improvements for most of the organizations becomes "a path full of obstacles and always away from the original path” [10, 11]. It is due to the fact that process improvement initiatives are not successfully implemented [12] or have limited success [13]. The main problem is the difficulty that an organization faces when adapting the selected process improvement model to their current scenario [10].

The goal of this paper is to present an organizational process improvement methodology overview. This methodology will enable a smooth and continuous improvement depending on their business goals, but allowing users to establish their improvement implementation pace. This will prevent initial resistance to change in the organization and the subsequent problems. Besides, the methodology is focused on the process improvement basic component known as “best practice”. Authors such as Fragidis and Tabanis [14] mention that the best practices are a critical factor to increase an organization’s process capacity and to achieve a competitive advantage. Therefore, this investigation tries to answer the followings questions:

- Why do actual software process improvement models and standards not have the expected performance when they are implemented in organizations?
- Why do improvements initiatives not have the expected results?
- Will the identification of organization’s best practices allow a process improvement initiative to be successfully implemented?

This paper is structured as follows: section two shows the research work context, section three describes the proposed methodology, section four shows a case study and finally section five shows the conclusions.

2 Research Context

The proposed methodology covers process improvement, best practices, knowledge management, change management and multi-model environment. Each of the topics covered is described briefly.

2.1 Process Improvement

Software Process Improvement (SPI) is a field of research and practice, arising out of the need to solve software development issues, increasingly complex and ubiquitous [15]. SPI is the action taken by organizations to change processes, taking into account the business needs and achieving their business goals more effectively [16].

SPI has become the primary approach for improving software quality and reliability, employee and customer satisfaction, and return on investment [17]. Different paradigms such as Quality Improvement Paradigm (QIP) [18] and methods and models for assessing the current capacity and/or maturity states of the organizations (like SCAMPI [19], ISO/IEC 15504 part 2: Performing an assessment [20]) or for implementing process improvement initiatives (like ISO/IEC 15504 [20] or IDEAL [21]) have been developed. However, implementing the methods above mentioned remains a high risk in many organizations.

According to Pries-Heje et al. [22], investments in SPI often have not achieved the expected changes and improvements. This is because the protocol followed by current process improvement is: 1) analyze the company weaknesses, 2) design their processes, often based on international models and standards, beyond the activities carried out in the company, and 3) implement the processes developed within the organization in depth (too many changes at the same time that are too great or too rapid) [23].

This way of implementing a process creates a process revolution, generating change resistance that leads to a failure in the process improvement initiative.

2.2 Best Practices

Williams [6] refers to best practices as building blocks of organizational learning and organizational knowledge but, what is a best practice?

According to Whiters [24], “a best practice could be a management or technical practice that has consistently demonstrated to improve one or more aspects such as productivity, cost, schedule, and quality or user satisfaction”.

In this context, using best practices in the organization’s software processes is a key element for improving the quality and the productivity, so that analyzing them can benefit processes at the speed supported by the organization [25, 26].

Due to the importance of the best practices, relevant institutions such as Software Engineering Institute (SEI), Project Management Institute (PMI), Institute of Electrical and Electronics Engineers (IEEE), and the International Organization for Standardization (ISO) have focused on the study of best practices. They have developed best practices reference models and standards [27].

The most widespread models and standards developed by these institutions are: CMMI-DEV v1.2, TSP, PMBOK, COBIT, PRINCE2, ISO/IEC15504, ISO9001:2000, ISO/IEC12207 [28].

However, although a wide variety of models and standards have been developed, West [23] highlights the current problem organizations experience when identifying their best practices, which is becoming one of the main challenges facing the new processes generation.

Successfully identifying an organization’s best practices allows the organization to have a base related to its strengths when developing an improvement initiative.

2.3 Knowledge Management

According to Burke and Howard [16], Knowledge Management (KM) is a systematic approach to facilitate the flow of data, information and knowledge to the right people at the right time, so they can act more efficiently and effectively. In this way, its importance for creating a value and building a competitive edge in organizations is well recognized.

On the other hand, according to Williams [6], KM allows the capture, codification, use and exploitation of the knowledge and experiences to develop better tools and methods, as well as to develop a willingness and ability to use these methods.

Therefore, KM requires an organizational effort to build, operate, maintain, and spread a knowledge–sharing environment. The organization by itself should be able to: 1)retrieve and understand the structured and unstructured data, 2)convert data into useful information and 3)share the knowledge [16].

2.4 Change Management

All improvements imply changes, unfortunately most of change initiatives fail because the inability to support the change is widely repeated despite the substantial resources dedicated to the effort, the people with necessary skills involved and everything supporting it [2].

Besides, it is necessary to take into account that changes scope, their competition with others or their speed can overwhelming the organization, destroying its investment in learning represented by organizational process assets and generating a degree of organizational tension difficult to manage that becomes a barrier to change [28].

Therefore, change management is the process of planning, organizing, coordinating and controlling internal and external components in order to ensure that process changes are implemented with the minimum deviation compared to approved plans and overall changes introduction goals.

2.5 Multi-model environment

People experience fear and dread due to common failures model-based process improvements [29]. As a result, organizations throughout the world are turning to an integration of international standards and models, in their effort to achieve a successful software process improvement [30, 31].

However, the difficulty in implementing process improvement successfully in multi-model improvement environments is well known. A first step in integration is to recognize that despite the different structures and terminologies, and despite different levels of abstractions, the standards and models used in the organization share common element types. The challenge is to examine the models to identify the common elements and then to organize implementation of these standards and models in the organization's processes in relation to these common elements [32, 28].

According to [33], the aims of a multi-model transformation are: reduce redundancy, improve integration, create synergy, leverage best practices, and make frameworks transparent. Therefore, a multi-model process transformation is characterized by a harmonized and unified approach to process improvement through implementation of multiple models [31].

A multi-model process transformation may contain three common elements: best practices elements, improved methods or institutionalization elements [34, 31].

3 MIGME-RRC Methodology

The purpose of this research work is to develop a methodology whose goal is to incorporate elemental process improvement components. The methodology will enable a smooth and continuous improvement depending on their business goals but allowing users to establish their improvement implementation pace. This will prevent initial resistance to change in the organization and the subsequent problems.

This methodology uses a bottom-up approach to software process improvement which consists in first identifying the internal best practices and promoting their use.

The best practices reinforce the organization's learning by documenting practices which have good results and promote their use so that the key organizational knowledge is preserved and transmitted. Then, the external best practices proposed by the most widespread models and standards through a multi-model environment, which complement the current practices and are suitable to the culture of the organization, could be included.

Therefore, it should be enable a smooth and continuous improvement of the capacity level of the organization's processes, depending on their business goals, but allowing users to establish their improvement implementation pace. In this way it will avoid the initial resistance to change.

Figure 1 shows the methodology scheme called *MIGME-RRC*: methodology for a gradual process improvement to reduce change resistance.

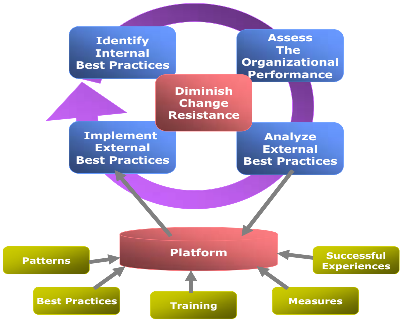

Identify Internal Best Practices
Assess The Organizational Performance
Diminish Change Resistance
Implement External Best Practices
Analyze External Best Practices
Platform
Patterns
Successful Experiences
Best Practices
Training
Measures

Figure 1. MIGME-RRC methodology

4 *Case Study*

4.1 Methodology Development and Implementation Environment

everis is a multinational consulting firm with factories in Europe and Latin America where they develop and implement the best practices to improve the performance of organizations. Its offer services which provide solutions to large companies in any sector and is based on three pillars: innovation, methodologies and efficiency through the use of specialists. The specialists use specific knowledge for each project and productivity in order to optimize time and cost.

Since its creation in 1996, it has grown both in invoicing and staff in a constant and always organic way. Its 2008 turnover was over 300M€ and more than 6000 employees, and the number of projects that remains open every mother is greater than 1000.

Christie and Fisher in [35] mention that, to successfully implement a process improvement in this kind of organizations, it must deal with a dynamically change conditions (in terms of growth, personal turnover and product evolution), in order not to obtain a series of unpredictable changes with unforeseen consequences.

Besides, there is the risk of incorporating deep changes (too many changes at the same time being carry out too rapid) based on external process implementation, external to the organization. These changes are accepted neither the organization nor the personnel. Therefore, the resistance is caused by the uncertainly and lost of control that they assume the changes will produce.

4.2 Case Study: Implementation on Project Management Processes Improvement

4.2.1 Scope

everis project management focuses on the scope of methodology implementation. Project management has a broad impact on their business goals and is measured in accordance with a series of objectives. Table 1 shows the most representative indicators related to the objectives that have been measured over the 2007-2008 period.

Table 1. **Indicator measured by everis over the 2007-2008 period.**

Indicator and Percentage
The percentage of management rules which are not right and not approved by the customer must be no more than 5%
The percentage of project planning that are not current and feasible must be no more than 5%
The percentage of start-up minutes that are not right and not approved by the customer must be no more than 5%

It is considered important to give a brief description of the indicators in order to understand their importance: 1) the management rules are a key document in the methodology because they define a project development framework, its detail planning and its management procedures; 2) the project planning documentation must be accurate and up-to-date at all times in order to have a proper overview of the project and, if necessary, to take the right decisions and, 3) the start-up minutes in order to come to an agreement with the client on the operational aspects of the projects.

It is important to highlight that the required level is very high; for example if the plan is not updated or the management rules are not approved by the client then the related indicator is not satisfied.

The implementation of the methodology focuses on project management carried out by account managers because: 1) they carry out the project management (heading one or more projects), and 2) they are in charge of projects. Besides, managers and project managers have a significant role in the best practices identification and validation feedback.

4.2.2 Identify Internal Best Practices

The aim of this phase is to analyze what practices are really carried out in the organization. If the practices are really identified, it will be possible to get a current process overview.

Figure 2 shows the analysis carried out in the organization, in order to identify the current practices.

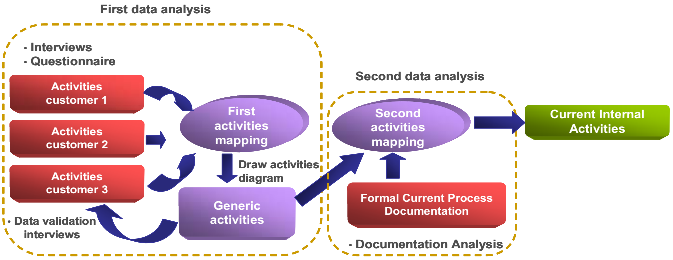


Figure 2. Best practices identification steps

The followed steps are:

- Conduct interviews with the organization’s managers.
- Analyze the information gathered from interviews, make activities diagrams for each interview and validate them with the accounting managers interviewed.
- Map the activities of all approved activities diagrams, in order to get a set of activities called “generic activities”. Then make a diagram with the generic activities.
- Analyze the formal processes documentation.
- Map the generic activities diagram with those included in the formal organizational processes documentation. It allows identifying the current organization’s activities.

4.2.3 Assess Organizational Actual Practices Performance

Once the current practices are identified, the organizational performance should be assessed in order to know how the organization's performance in relation to the current practices. Figure 3 shows how the organizational performance was analyzed; the current activities, business goals and key indicators are used as input source.

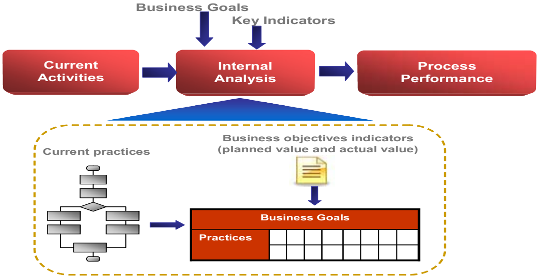


Figure 3. Organizational performance asses

The steps followed are:

- Collect and analyze any formal documentation about planned values of indicators related to organizational business goals.
- Collect actual values of the indicators.
- Make a matrix of business goals indicators and generic activities.
- Analyze the achievement of business objectives according to the planned value and the matrix, and establish a performance overview.
- Prioritize the business goals to be achieved.

In the case of everis, internal control audits where taken as input source, Figure 4 shows the percentage of business objectives everis achieved with the current practices. The analysis of the indicators will allow the project management to make an internal project characterization.

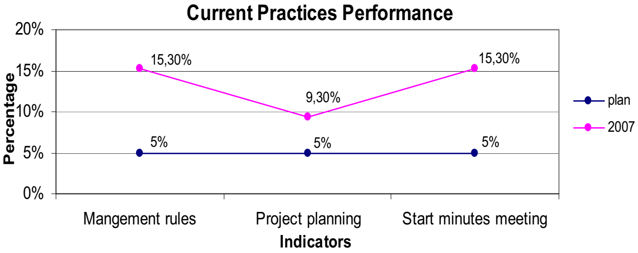


Figure 4. everis audits results

The analysis of everis audits allows that an overview of the key indicators achieved with the current practices to be established. Although the three percentages of the indicators analyzed are not too far from the planned percentage, all of them should be improved, due to its importance for achieving the business objectives.

4.2.4 Analyze External Best Practices

In this phase, the performance of similar models and standards were analyzed in order to establish a multi-model environment. The models and standards included in the study are: CMMI-DEV v1.2, PMBOOK, PRINCE2, TSP, COBIT, ISO9001, and ISO/IEC 15504. The steps followed are [27]:

- Select the models and standards to be analyzed.
- Choose the reference model.
- Select the process.
- Establish the detail level.
- Create a correspondence template.
- Identify the similarity among models.
- Show the obtained results

The tables which result of the study contain the practices candidates to be included. So that, the study indicates what project management external best practices, where performance is proved, could be executed by the organization.

4.2.5 Implement External Best Practices

This phase includes analyzing what and how the additional best practices would be incorporated depending on their dependence and importance in achieving a business goal. The steps followed are:

- Identify the change resistance factors and the improvement initiatives risks, because in many cases, both are important factors that make difficult an improvement initiative implementation and lead it to the failure.
- Analyze the external best practices not yet implemented, having in count practices dependences and its impact on achieving business goals.

Incorporate additional best practices from a multi-model environment. For this selection and implementation it should be enabled a set of best practices, which allow the users to select them in a smooth and continuous way. These best practices always will be selected in a pace established by them.

In the special case of everis, the improved processes (that contains the best practices identified and a set of external practices to be incorporated) were grouped in a project management method, because of the need of the organization to develop a project management method as a part of its COrporate Methods methodology (COM).

Figure 5 shows the process improvement implementation through the project management method overview. The method gathers all the everis knowledge, experiences and best practices. Besides, when enabling a set of best practices it will be possible to establish a marked-up user environment for improving processes continuously.

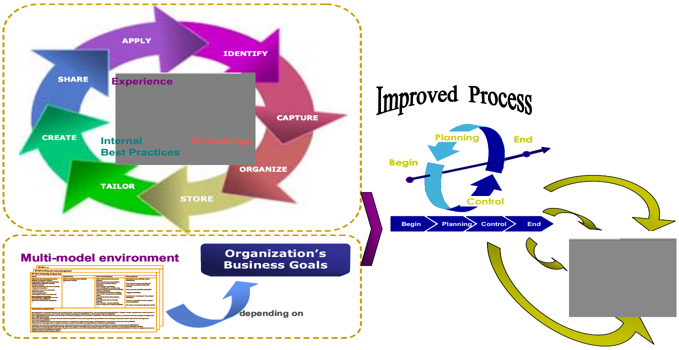


Figure 5. Process Improvement Implementation through COM Project Management Method

Once the method was developed, it was validated and approved by the quality and methodology everis group. Then pilot projects were performed. The aim of the pilot projects was to launch a better version of the COM project management method. The launch allowed testing the method in the execution of the project and after analyzing their results, deploying them as an experience of using best practices.

Using pilot studies reduces the risk of rolling out a flawed process, procedure or other solution component. Besides, experiences in pilot studies always expose improvement opportunities that can be exploited to tune and refine the improvement process before a broader dissemination [26, 36, 37].

The chosen pilot projects features in order to reduce risk in implementing the method were: 1) size medium (duration time not greater than 3 months); 2) Budget: 100-150000€; 3) Staff: 4-7 people; and 4) Project manager profile (begin as a project leader).

Meetings with the projects managers were held in order to gather information about the improvement. During the interviews, their comments on project management method acceptance and usefulness in the project management were gathered.

4.2.6 Improvement Measurement and Results

Three measures were defined in order to analyze the improvement: process use (M1), process performance (M2), and process acceptance (M3). The approach involves gathering information: before, during and after the change.

This approach allows getting better improvement performance information. According to Kasunic [26, 36], typical approaches that fail are those whose measurable observations are done before the change or after the change.

Process use (M1): Its aim is to analyze the degree of best practices performance. This measure data were gotten from the Madrid office internal audits. Figure 6 shows the results obtained by comparing the planned percentage against actual percentage in 2007 and 2008. In 2008, an improvement was observed in the project planning practices and start-up minutes indicators; both were 5%.

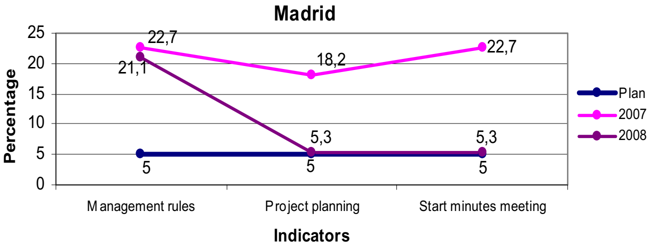


Figure 6. Madrid office best practices performance

Process performance (M2): Its aim is to analyze the process performance obtained with the improvement. This measure data were gotten from Madrid office delivery projects. Figure 7 shows the obtained results by analyzing those who have any type of internal cost deviation (either in incurred hours, external costs or subcontracting). However the deviations did not affect the time limits agreed with customers.

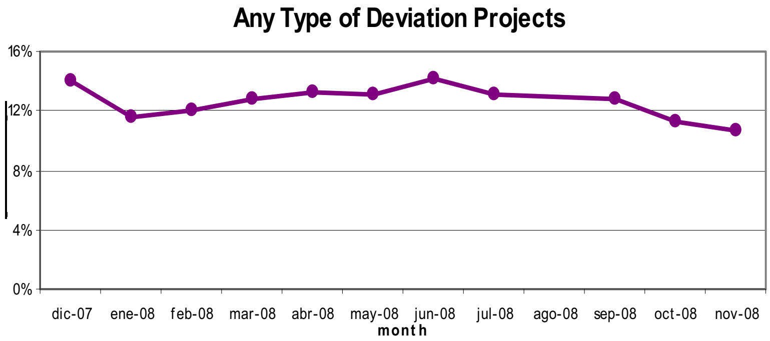


Figure 7. Projects with internal deviations

An improvement in delivery projects which have any type of deviation was observed because the percentage of projects with deviations decreased from 13.99% to 10.63%.

Process acceptation (M3): Its aim is to analyze the process acceptation by users and, therefore, it allows checking the reduction in change resistance. This measure data were gotten by analyzing surveys carried out by Madrid office users involved in project planning.

Figure 8 shows two graphics with improvement acceptance results obtained. In graphic 8(a), an improvement in the use of the COM methodology is observed (from 10% in 2007 to 31% in 2008). In graphic 8(b), an improvement in the usefulness of the COM methodology is observed (from 10% in 2007 to 95% in 2008).

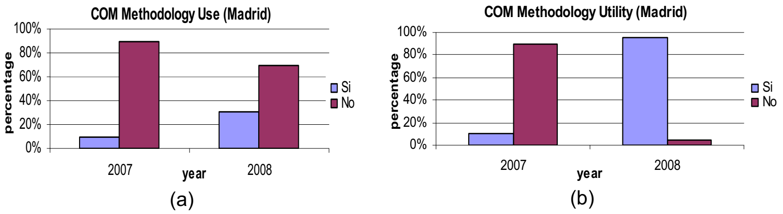


Figure 8. Improvement acceptation by users

5 Conclusions

The implementation of external processes in the organization is the main cause of process improvement initiatives failure. This involves a high-cost of assimilating the appropriate knowledge, big cultural change, and its process deployment.

MIGME-RRC, shows that beginning a process improvement initiative by identifying the organization's best practices, involving the relevant stakeholders (chosen by the organization's senior management as success stories), has allowed extracting, collecting and formalizing the tacit knowledge of the organization.

Having the knowledge in a formal way, it is possible to select a set of best practices based on the way the organization's works, but in accordance with their business goals. It reduces the staff's resistance to change when implementing process improvement initiatives.

Besides, the reduced change resistance was achieved because of a smooth and continuous best practices implementation, and most of all, the changes are introduced by users and, therefore, approved by relevant stakeholders.

The methodology allows the different units of a software intensive organization to set their improvement pace and choose those practices that best fit their work, thereby doing it more efficient. At the same time, there is a uniform process capacity through the organization. Therefore, the communication of improvement results and their justification is a key point.

Finally, the implementation of MIGME-RRC for a smooth and continuous improvement performed in everis confirms that people only accept assimilated changes with identified benefits. In this way, they perceive the change as an evolution of they work.

Acknowledgements. This work is sponsored by Endesa, everis Foundation and Polytechnic University of Madrid through the Software Process Improvement Research Group for Spain and Latin American Region. Deserves special mention the support provided by everis providing data and allowing the realization of the pilots, without whose help had not been possible the realization of this investigation.

6 Literature

1. Persse, J., Project Management Success with CMMI, (eds.) Prentice Hall, pp. 321 (2007)
2. Senge, P., Kleiner, A., Roberts, Ch., Ross, R., Roth, G. & Smith, B., La Danza del Cambio: Cómo Crear Organizaciones Abiertas al Aprendizaje, (eds.) Gestion 2000, S.A., pp. 11-17 (2000)
3. Pressman, R., Ingeniería del Software Un Enfoque Práctico, (eds.) McGraw-Hill, sexta edición, pp. 640-662 (2006)
4. Walker, R., Software Project Management A Unified Framework, (eds.) Addison-Wesley, pp. 5 (1999)
5. Yager, S., Top 10 Project Management Problems, CEO Artemis International, http://dssresources.com/papers/features/yager10032001.htm (2001)
6. Williams, T., How Do Organizations Learn Lessons From Projects—And Do They? Engineering Management, IEEE Transactions on, Volume 55, Issue 2, pp.248 – 266, Digital Object Identifier 10.1109/TEM.2007.912920 (2008)
7. Gainer, J., Best Practices: Informal, Effective Process Improvement, Jeff Gainer's CRITICAL PATH, http://www.itmweb.com/f041598.htm (1998)
8. Software Engineering Institute (SEI), CMMI Performance Results TATA Consultancy Services, Software Engineering Institute, Carnegie Mellow University, http://www.sei.cmu.edu/cmmi/results/org29.html#BC2 (2008)
9. Gibson, D., Goldenson, D., Kost K., Performance Results of CMMI-Based Process Improvement, Technical Report CMU/SEI-2006-TR-004 ESC-TR-2006-004, Software Engineering Institute (2006)
10. Morgan, P., Process Improvement- Is it a lottery? Software development magazine: UML, Agile, programming, software testing, quality assurance and project management, http://www.methodsandtools.com/archive/archive.php?id=52 (2007)
11. Potter, N., Sakry, M., Making Process Improvement Work: A Concise Action Guide for Software Managers and Practitioners, Ed. Addison-Wesley, pp. 2-5 (2006)
12. Goldenson, D., Teach Views, Performance Outcomes from Process Improvement, SoftwareTechn news, Vol.10 No. 1 (2007)
13. Conradi, H.; Fuggetta, A., Improving software process improvement, Software, IEEE, Volume 19, Issue 4, pp.92–99, Digital Object Identifier 10.1109/MS.2002.1020295 (2002)
14. Fragidis, G. & Tarabanis, K., From Repositories of Best Practices to Networks of Best Practices, IEEE, ISBN 1-4244-0148/06 (2006)
15. Kautz, K., Levine, L., Hefley, B., Johansen, J., Kristensen, C., & Nielsen, P., Networked Technologies - The Role of Networks in the Diffusion and Adoption of Software Process Improvement (SPI) Approaches, Software Engineering Institute (2004)
16. Burke, G., Howard, W., Knowledge Management and Process Improvement: A Union of Two Disciplines, CrossTalk the journal of defense Software Engineering (2005)
17. Mathiassen, L., Ngwenyama, O.K., Aaen, I., Managing change in software process improvement, Software, IEEE, Volume 22, Issue 6, Digital Object Identifier 10.1109/MS.2005.159 (2005)
18. Basili, V., Caldiera, G., and Rombach, H. D. (2002) The Experience Factory, https://wwwagse.informatik.uni-kl.de/pubs/repository/basili94c/encyclo.ef.pdf Accessed 28 Oct 2007

19. Members of the Assessment Method Integrated Team, Standard CMMI® Appraisal Method for Process Improvement (SCAMPI), Version 1.1, Handbook CMU/SEI-2001-HB-001, Software Engineering Institute, Carnegie Mellon University (2001)
20. AENOR, ISO/IEC 15504: 2004 Information technology – Process assessment, first edition 2004 (2004)
21. McFeeley, B. IDEALSM: A User's Guide for Software Process Improvement, Software Engineering Institute, Handbook CMU/SEI-96-HB-001 (1996)
22. Pries-Heje, J., Christiansen, M., Johansen, J., y Korsa, M., The ImprovAbility Model, CROSSTALK the Journal of Defense Software Engineering (2007)
23. West, D., Enough Process Let's Do Practices: An Introduction to EssWork and EssUP, Information Technology (IT) Conference 2007 (2007)
24. Withers D., Software Engineering Best Practices Applied to the Modeling Process, Proceedings of the 2000 Winter Simulation Conference (2000)
25. Beynon D., Interpreting Capability Maturity model Integration (CMMI) for business Development organizations in the Government and Industrial Business Sectors, Software Engineering Institute, Technical Note (2007)
26. OSD Comptroller iCenter, Best Practices & Benchmarking: Making worthwhile comparisons (2004)
27. Kasunic Marck, An Integrated View of Process and Measurement, Software Engineering Institute (2003)
28. Calvo-Manzano, J. A., Cuevas, G., Muñoz, M., San Feliu, T., Process Similarity Study: Case Study on Project Planning Practices Base don CMMI-DEV v1.2, EuroSPI 2008 Industrial Proceedings, ISBN 978-87-7398-150-4 (2008)
29. Mogilensky, J. (2009). Pathological Box-Checking: The Dark Side of Process Improvement. SEPG 2009 Conference.
30. OST SEPG (2009). CMMI, ISO and ANSI 748 High value and easier than you think. SEPG Conference 2009.
31. Srivastava, N., Singh, S. and Dokken, T. (2009). Assorted Chocolates & Cookies in a Multi-model Box. SEPG 2009 Conference.
32. Kirwan, P., Andelfinger, U., Sassenburg, H., and Heijstek, A. (2004). A Unified Process Improvement Approach for Multi-model Improvement Environments. Software Engineering Institute (SEI). Carnegie Mellow University. http://www.sei.cmu.edu/news-at-sei/features/2006/04/feature-1-2006-04.htm. Accessed 13 Nov 2007.
33. Lawrence, J. and Becker, N. (2009). Implementing Insights and Lesson Learned Using ISF for Excellence. SEPG 2009 Conference.
34. Siviy, J., LynnPenn, M. and Robert Stoddard. (2007). Achieving Success via Multi-Model Process Improvement. Software Engineering Institute (SEI). Carnegie Mellow University. http://www.sei.cmu.edu/sema/presentations/multi-model.pdf. Accessed 13 Nov 2007.
35. Christie, A. and Fisher, D., Simulating the emergent behaviour of complex software-intensive organizations, ProSim 2000 Workshop (2000)
36. Kasunic, M., Conducting Effective Pilot Studies, 2004 SEPG Conference (2004)
37. Fowler, P. & Rifkin, S., Software Engineering Process Group Guide, Technical Report CMU/SEI-90-TR-024 ESD-90-TR-225, Software Engineering Institute (1990)

7 Author CVs

Calvo-Manzano Jose A.

He is PhD in Computer Science. He is assistant professor in the Computer Science School at the Polytechnic University of Madrid. He is teaching in the area of Software Engineering, specifically in the domain of software process management and improvement. He has participated in more than 20 research projects (European and Spanish Public Administration). He is author of more than 50 international papers. He is author of books related to software process improvement and software engineering topics also. He is a member of SOMEPRO research group, where his main research field is process improvement, focusing in the topics of requirements management, project management, process deployment, and solicitation and supplier agreement development.

Cuevas Gonzalo

He received an Engineering degree in Telecommunications in 1965 and a PhD in Telecommunications in 1974. He also received an MS in Computer Science from the Polytechnic University of Madrid in 1972. He has been vice dean of the Computer Science faculty at the Polytechnic University of Madrid, where he is a full professor since 1970. He worked for Iberia Airlines as Software Development Director, supervising over 200 technicians, Data Processing Centre Director, Data Transmission Software Development Director, and the person in charge of strategic planning and software process improvement. He led European projects on software best practices from 1991 to 1995. His main research field is software engineering, including both technology (methods, techniques, and formalisms) and management. His current research interest is process models and methods, and transition packages. He is member of ACM, senior member of IEEE, member of the Telecommunication Engineering Association and member of the Computer Sciences Association.

Gómez Gerzon

He is a B.Sc. in Computer Science, Autonomous University of Tamaulipas (UAT), Mexico. He was assistant professor in the same university. He has occupied many management positions during more 4 years in the industry. He has published several technical papers on IT Service Management. Nowadays, he is a PhD student in the Faculty of Computer Science of the Polytechnic University of Madrid, Spain. He works in a “Metamodel for Service Level Management”.

Mejia Jezreel

He is a Master in Computer Science (2003-2005) and Computer Science Engineer (1996-2000) from the Institute Technological of Orizaba University of México. He is actually Software Engineering PhD student from the Computer Science faculty at the Polytechnic University of Madrid. His research field is Software Engineering. His current research interest consists on outsourcing process improvement. He has published several technical papers on acquisition process improvement. He is colaboring with an everis Foundation research project.

Muñoz Mirna

She is a Master in Computer Science (2002-2004) and Computer Science Engineer (1996-2000) from the Institute Technological of Orizaba University of México. She is actually Software Engineering PhD student from the Computer Science faculty at the Polytechnic University of Madrid. Her research field is Software Engineering. Her current research interest consists of the development of a methodology whose goal is to incorporate process improvement focusing on diminishing change resistance. She has published several technical papers on project management and process improvement. She is colaboring with an everis consultants research project.

San Feliu Tomás

He is PhD in Computer Science. He has been working 15 years to Software Engineering field as programmer, associate and assistant professor and consultant. He has been involved in CMM appraisals in Spain and Latin America. As Software Engineering assistant professor at the Polytechnic University of Madrid, his main research field is software process and software process improvement. His current research interest includes project management and risk management. He has published several books on process improvement and he has published technical papers in software engineering and software process improvement.

Sánchez Ángel

Ángel Sánchez was born in Madrid where he obtained his B.S. in Physics by the Complutense University and Ph.D. in Applied Physics by CSIC (Spanish National Research Council). After leaving research, he joined IBM where he was in charge of Quality and Environ-mental Management software-based systems. In 1998, he moved to everis, having different responsibilities in IT Governance and Quality Management. Now he is enrolled in the Innovation Department as a Development Director, in charge of identifying high potential technological opportunities for everis. He has published several articles in the field of Quantum Optics, Photonics and IT Governance.

A Dynamic Model for Improving Knowledge Sharing in Virtual Organisations

Elli Georgiadou[1], Kerstin V. Siakas [2]

[1] *Middlesex University, School of Engineering and Information Sciences,*
The Burroughs, London NW4 4BT, E-mail: e.georgiadou@mdx.ac.uk

[2] *Alexander Technological Educational Institution of Thessaloniki,*
Department of Informatics, P.O. Box. 141, GR-57400, Greece
Email: siaka@it.teithe.gr

Abstract

Knowledge Sharing is by far the most important component of a Knowledge Management programme. Organisations strive to gain competitive advantage through efficiencies. When organisations realise that organisational knowledge is by far the most valuable resource they need to find ways for efficient and effective Knowledge Sharing. We propose the dynamic Knowledge Acquisition and Sharing Lifecycle (KASL II) model for aiding the knowledge sharing process through showing the stages of translating an organisation's mission and goals into objectives, and how decisions and actions operate for materialising these objectives.

Key words: Knowledge Management, Knowledge sharing, learning, virtual teams, KASL

1 Introduction, Motivation and Perspectives

In today's highly competitive and rapidly changing global environment more and more organisations strive to form virtual teams comprised of experts situated in different locations, organisations, countries and time zones. The increased complexity of international organisations and worldwide business relationships has become a dynamic business reality with intensified competition. Outsourcing and distributed teams that seldom meat in reality is common practice today. Teamwork is essentially a result of human interaction. Virtual teams are teams of people who primarily interact electronically and who may meet face-to-face occasionally and in some projects not at all. In a virtual team the team members work interdependently towards a shared goal by webs of Information and Communication Technologies (ICTs) across time and space and often across organisational boundaries (Handy, 2000; Lipnack and Stamps, 1997; Malhotra, 1996; Mansour-Cole, 2001).

Despite the many technologies that support collaboration among distributed work groups, organisations still face difficulties building online work environments. What is lacking in most virtual workplaces is a proven methodology for identifying and converting individual expertise, skills, and experience into organisational knowledge and to strategically align organisational knowledge transfer and learning investment with organisational value outcome.

Harorimana (2006) argues that it is impossible to transfer knowledge that is not embedded in local cultural practices and settings because reciprocity norms dominate successful knowledge transfer. We believe that it is only a challenge we can face by raising cultural awareness. By sharing information across the organisation, virtual teams naturally build their own knowledge bases that are consistent with the rest of the company. The ideal environment and working practices will be to change the mindset and behaviour of team members so that instead of perceiving knowledge sharing as an extra task for the team members, isolated from the knowledge of other team members, it (knowledge sharing) becomes the natural way to work for everyone. The result will be a well-integrated, highly responsive organisation whose employees can quickly take action regardless of location.

In this paper we propose a knowledge acquisition and sharing lifecycle for use in virtual organisations. The application of the model to everyday processes will ensure that the output of every team adheres to the company's overall strategy.

2 Knowledge Management concepts

Knowledge Management (KM) can be defined as the management of "*processes by which knowledge is created and applied*" (Paulzen et. al, 2002), though there is not a commonly agreed definition. KM can be viewed as the process of turning data into information (data in context) and, further on, to knowledge (use of information) (Kanter, 1999; Spiegler, 2000) or as the organisationally specified systematic process for acquiring, organising and communicating both tacit and explicit knowledge of employees so that other employees may make use of it (knowledge sharing) in order to be more effective and productive (Alavi and Leidner, 2001). Tacit or implicit knowledge is context-specific, personal and subjective including cognitive elements and thus difficult to formalize and communicate (Davis et. al, 2005; Siakasand Georgiadou, 2006).

KM is a business philosophy. It is an emerging set of principles, processes, organisational structures, and technology applications that help people share and leverage their knowledge to meet their business objectives (Gurteen, 1999). This focuses the individual and places responsibility on the individual, the knowledge worker. At the same time KM programmes in organisations emphasise the holistic nature of creating, sharing and managing knowledge.

Knowledge sharing (transfer) is the process where individuals mutually exchange both tacit (feel or sense for something (Kautz and Kjaergaard, 2008)), and explicit knowledge (codifiable knowledge (Kautz and Kjaergaard, 2008)), and jointly create new knowledge. This process is essential in transferring individual knowledge into organisational knowledge. The capability of an organisation to create, recognise, widely disseminate and embody knowledge in new products and technologies is critical when faced with turbulent markets, high competition and financial instability (Nonaka, 1991). Continuous knowledge creation requires voluntary actions including openness, scrutiny, trust and tolerance towards different views and interpretations. Organisations expect employees to keep professionally up-to-date by continuously obtaining internal and external information relating to their profession. Knowledge evolves continuously as the individual and the organisation adapt to influences from the external and the internal environment. Elron and Vigoda-Gadot (2006) found that when ICTs are used as the main communication channel between team members the limitations of the communication increase, as technology cannot provide the same richness as face-to-face interactions and potentially hinder the effectiveness of knowledge sharing. They also found that influence tactics and political processes in virtual teams are more restrained and mild than in face-to-face teams. This seems to indicate that bottom-up empowerment should be encouraged to improve interaction and communication richness.

2.1 KM and ICTs

Organisations are facing a new challenging environment characterised by globalisation, dynamism and increasing levels of complexity due to rapid changes in technology and its connected intricate knowledge.

KM plays an important role in software development (Kautz and Thauysen, 2001). The literature emphasizes mainly implementation of new Information Technology (IT) systems and technical solutions. Organisational and cultural aspects are usually neglected. Organisations formally capture, manage and store explicitly knowledge with the help of computer-based systems, such as

Management Information Systems (MIS), Decision Support Systems (DSS) and Expert Systems (ES), which today are becoming ubiquitous in organisations (Davenport and Prusak, 1998). However technology by itself often does not solve an organisation's inherent problems relating to intellectual capital, knowledge and information management. Davis et. al (2005) argue that KM is based to only 30% on implemented systems and the rest on people. The fact is that the view of knowledge is changing and today it is seen as human capital that 'walks out the door at the end of the day' (Spiegler, 2000). ICTs seem to enhance the KM capabilities of organisations (Alavi and Leidner, 2001; Tanriverdi, 2005).

Internet-based virtual tools have created new opportunities for rapid access to business information world-wide. Identifying potential business partners and developing business links with organisations in other countries has become easier for organisations that are experienced in monitoring web-based information sources, and are able to combine tacit knowledge with new knowledge sources that are enabled by ICTs, such as internet, intranet, groupware and Computer Supported Co-operated Work (CSCW) systems. Explicit knowledge is transferable through formal and systematic languages. Organizations try to gain business advantage by using Knowledge Creation processes (KC) in order to "capture" knowledge and use it to make wiser decisions about strategy, competition, products, production and service life cycles (Davenport and Prusak 1997), as well as to improve its effort in today's very competitive and uncertain environment. Organizational Knowledge is created by an organizationally specified systematic process for acquiring, organizing and communicating both tacit and explicit knowledge of employees so that other employees may make use of it in order to be more effective and productive (Alavi and Leinder, 2001). This experience is documented and stored in a Knowledge Management System (KMS) preparing the organization to react on the future, based on the Knowledge that is acquired from its own organisational experience.

Views on Knowledge Management (KM) and ICTs are wide ranging between two poles - one considering the relationships between KM and ICT incidental – the other considering Information Technologies (IT) being the core of KM (Holsapple, 2005). This paper considers KM being a social and human phenomenon which by using ICT as a tool can improve the efficiency of knowledge creation, visualisation, transfer and preservation. ICTs facilitate the amplification, augmentation and leverage of innate human knowledge handling capabilities. The advances in ICT provide organizations with increased flexibility and responsiveness, permitting them to rapidly form dispersed and disparate experts into a virtual team that can work on an urgent project. ICTs support faster, cheaper and more reliable knowledge work of large scale and the existence of efficient ICT is inevitable an imperative requirement for the existence of virtual collaboration. However, the emphasis in this paper is to unfold the human and cultural challenges that can create added competitive value for virtual and networked organisations.

2.2 KM - Communities of Practice and Social Computing

Communities of Practice (CoP) are defined by Lave and Wenger (1991) as "an aggregate of people who come together around mutual engagement in an endeavour" and by Bettoni et al. (2006) as "the participative cultivation of knowledge in a voluntary informal social group". The highlight in both definitions is on a type social construction or community leading to a kind of culture including common practices that emerge in the course of the mutual endeavour. The community is usually born around a shared profession and its topics of discussion outside of the traditional structural boundaries. However, both experience and research show that our knowledge for designing online CoP is limited (Barab et al., 2004). Some researchers even claim that enthusiasm about CoP is well beyond empirical evidence (Schwen and Hara, 2003). In fact, many communities lack sustainability by falling apart soon after their initial launch due to lack of enough energy and synergies or by adopting a short-term opportunity driven behaviour, which both in turn leads to uncertainty and mistrust between the members and consequently to low quality of shared work results (Bettoni et. al, 2006). The benefits of CoP seem to include the facilitation of greater variety in the knowledge domains of the members (De Carolis and Corvello, 2006).

Social computing refers to the use of social software within networks for creating and maintaining mutual social connections among individuals (Kwai Fun IP and Wagner, 2007). Such contemporary networks are learning communities in the sense that they evolve through collective building and transfer of knowledge. The participation of members is shifting (Lave and Wenger, 1991). Social computing includes computer supported cooperative work and learning and is mediated through email, wiki (a collaborative technology, that allow for linking among any number of pages, for organising information on Web sites), blogs (a website where entries are written in chronological order and displayed in reverse chronological order) instant messaging, videoconferencing etc.

The potential role of social computing and Communities of Practice (CoP) enables a bottom-up approach for supporting knowledge creation and knowledge sharing activities in contrast to the hierarchical control of central knowledge repositories.

3 Utilising KM for improving the effectiveness of Virtual Teams

The characteristics of virtual teams identified by Bal and Teo (2000) are as follows: Virtual teams consist of goal oriented team members/knowledge workers, who are dispersed geographically and work supported by ICT more apart than in same location. They solve problems and make decisions jointly; they are involved in a coordinated undertaking of interrelated activities and are mutually accountable for team results. The virtual teams have usually a finite duration (few teams are permanent). The primary motivation is to gain access to world class capabilities to lower costs and to integrate diverse perspectives (Siakas and Balstrup, 2005). Virtual teams, by their very nature, imply the presence of a group of geographically dispersed individuals often from different cultural, educational and professional backgrounds. They work on a joint project or common task and communicate, mainly by using e-mail, for the duration of a specific project (Järvenpää and Leidner, 1999). A potential conflict arises when the team members belong to different organisational and cultural units, because the team mates do not know where to place their loyalty (Balstrup, 2004). In virtual environment this is exacerbated, because informal communication is reduced, due to the fact that members rarely meet face-to-face. A successful leader of a virtual team must excel in applying the right choice of ICT to enable effective communication and knowledge sharing. Communication, and thus also knowledge sharing, in virtual teams in a global context is considerable much more difficult due to language, culture, time issues and distance. Knowledge sharing with bad communication is a big challenge and a difficult issue to achieve. Teams lacking communication and knowledge sharing will turn into detached groups of uninvolved strangers out of leadership and cooperation. The individuals of the virtual team and the leader must build a unified team committed to the common goal and through interdependent interaction generate group identity and create the feeling of belonging to a group (Balstrup, 2004).

In today’s competitive environment increasingly large numbers of Information Technology (IT) organisations use virtual teams in their international operations, which can constitute subsidiaries, outsourcing relationships or global partnerships (Siakas and Balstrup, 2006). A sense of identity is important because it determines how an individual directs his or her attention (Wenger, 1999). Identity shapes what one pays attention to and constitutes a primary factor in learning and sharing of personal experiences (knowledge transfer). It is proved that strong identity within CoP contributes towards better collaboration, learning and innovation. However, individuals of virtual teams and communities of dispersed workers show difficulties in interacting with colleagues and keeping themselves up to date.

Social computing and CoP develop spontaneity for solving professional daily problems and can to some degree substitute informal discussions of collocated teams. Subsequently this kind of social networking is an important source for building trust, creating reciprocal esteem, as well as for developing a feeling of identity and group-belonging (Siakas and Siakas, 2008; Siakas and Balstrup, 2006). If the relationships and social rules are based more on professional than on personal or affective factors the social networking can constitute an important, yet often unrecognized, supplement to the value that individual members of a community obtain in the form of enriched learning and a higher motivation to transfer what they learn and in this sense even substitute formal teaching programs (De Carolis and Corvello, 2006). Also there is evidence asserting that CoP create organisational/institutional value (Storck and Hill, 2000; Wenger and Synader 2000). Social networks function at higher level of abstraction and contribute to a high degree to tacit knowledge sharing. We need to understand that CoP are governed by mutual benefit norms in which the community welfare takes priority over individual interests.

4 A Knowledge Acquisition and Sharing Lifecycle

Reflecting on the literature review regarding KM and personal experience from working in multicultural environments, as well as from teaching multicultural groups of students Georgiadou et al. (2006) developed a lifecycle for knowledge acquisition and sharing which they used initially in academia to model the knowledge processes involved in student group work. This model (KASL) is in this paper extended to encompass industrial situations which are almost always based on team work including work carried out by virtual, dispersed and diverse teams . Individuals learn and contribute to the group, the group learns and contributes to the organisation, and the organisation facilitates/sponsors the processes in a perpetual virtuous circle of exchange of information, sharing of knowledge and process improvement.

The vision refers to mental images of the future, which become tangible in the form of mission statements. The mission statements define the primary purpose and articulate the responsibilities to its stakeholders. Goals are attempts to improve performance by making mission statements more concrete. Objectives represent the operational definitions of goals in more precise terms and describe what needs to be accomplished in order to reach the goals. Plans and tasks are developed usually by managers to help accomplish higher-level intentions.

KASL–II (Figure 1) depicts four learning loops which involve individuals, groups, groups of groups (departments) and the whole organisation.

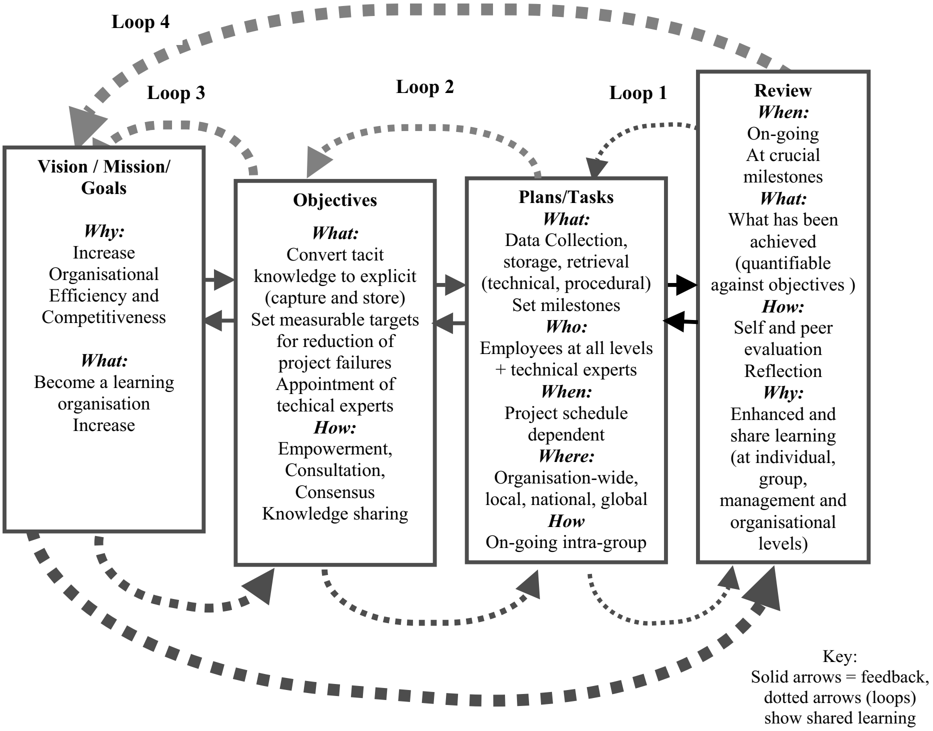

Loop 4
Loop 3
Loop 2
Loop 1
Vision / Mission/ Goals
Why:
Increase Organisational Efficiency and Competitiveness
What:
Become a learning organisation
Increase
Objectives
What:
Convert tacit knowledge to explicit (capture and store)
Set measurable targets for reduction of project failures
Appointment of techical experts
How:
Empowerment, Consultation, Consensus
Knowledge sharing
Plans/Tasks
What:
Data Collection, storage, retrieval (technical, procedural)
Set milestones
Who:
Employees at all levels + technical experts
When:
Project schedule dependent
Where:
Organisation-wide, local, national, global
How
On-going intra-group
Review
When:
On-going
At crucial milestones
What:
What has been achieved (quantifiable against objectives)
How:
Self and peer evaluation
Reflection
Why:
Enhanced and share learning (at individual, group, management and organisational levels)
Key:
Solid arrows = feedback, dotted arrows (loops) show shared learning

Fig. 1: KASL – II, A dynamic four loop model for knowledge sharing and learning

The KASL-II model depicts four knowledge sharing and learning loops reflecting the stages of translating an organisation’s vision and goals into objectives and the objectives to tasks. At each one of the stages feedback loops to the preceding stage ensure that omissions and problems are captured at the earliest opportunity, modifications to the schedule, resource allocation and quality monitoring are enabled through these feedback mechanisms. Knowledge is captured, stored and accessed for improved decision making. Measurable targets are set and monitored, hence the process is controllable and is likely to achieve maximum improvements.

Learning loops show the granularity of activities through detailed and systematic posing of relevant questions which need to be addresses at each stage.

Loop 1 shows the learning gained by individuals who engage in the tasks and activities (smallest granule). Here, individual employees have opportunities for self and peer assessment, reflection and reporting of measurable results.

Loop 2 shows the learning gained by groups (second level of granule) on groups of activities (parts of projects). Again feedback to the setting of objectives stage four learning loops which involve individuals, groups, management and the organisation.

Loop 3 shows the learning gained by larger groups such as departments/sections (groups of groups) where objectives are set, revised and assessed. This phase also encapsulates at the organisational level (granule) process management, process improvement, setting of measurable targets, prioritising objectives, allocating/reallocating resources and facilitating conflict resolution

Finally, loop 4 shows the organisational learning which is the vehicle for achieving the organisation's vision and goals. A learning organisation is able to reflect and capitalise on the achievement of targets which in turn enhance the organisation's competitiveness. When all the employees feel empowered and responsible for the process (they are involved with) and when they shed the old way of thinking by replacing the belief in knowledge sharing rather than in knowledge hoarding it (the organisation) will move from "knowledge is power" to "Shared Knowledge is power". Improvements in learning at all four levels moves an organisation from data handling, through to information, knowledge and wisdom ensuring the competitiveness of the organisation. Individuals feel valued and work for the benefit of the organisation which is no longer in conflict with their own ambition. As early as 1981 Enid Mumford identified the concept of knowledge fit, job satisfaction, technical fit and the benefits of this approach to everybody involved. Nearly 30 years later the Knowledge Management community is putting these ideas into practice.

The KASL II model aims to make the process of knowledge sharing and learning process explicit at all levels of granularity by going back to first principles of asking those "honest serving men" who according to Kipling (http://www.kipling.org.uk/poems_serving.htm last visited on March 13th, 2009) they taught him all he knew" …

I keep six honest serving-men
(They taught me all I knew);
Their names are What and Why and When
And How and Where and Who.
I send them over land and sea,
I send them east and west;
But after they have worked for me,
I give them all a rest.
(Rudyard Kipling, 1902)

In addition the KASL II model depicts the different dynamics involved in knowledge acquisition and knowledge sharing on four different abstraction levels, namely the individual, the group, departments (groups of groups) and the organisation. The organisation works towards realising its vision and achieving its mission. Objectives are achieved through consensus. Normally when the workforce is involved in setting the objectives they have ownership of the project and hence they work collaboratively. Individuals' tacit knowledge is externalised, shared and formalised (changed to explicit knowledge) initially with the direct collaborators (such as a project team). Different project teams share knowledge through integrated repositories. Thus the organisational knowledge grows all the time. The attitudes of the staff change from individualistic to collectivistic. The ‘enemy’ is the competition and hence it is not internal. Employees feel valued and secure in sharing their knowledge with their colleagues. Issues of Intellectual Property Rights (IPR), Exploitation Rights (ER), Ethics and Culture need to be addressed particularly as knowledge sharing takes place at intra-organisational and inter-organisational level and across national boundaries, national cultures and languages.

The knowledge cycle within organisations includes mechanisms of recording the ownership of knowledge, its capturing, organising, representing and storing, its retrieving and the creation of new knowledge. Within a learning organisation each employee becomes a knowledge worker. Organisational memory is valued and shared using management and technical tools. Appropriate techniques and tools for KM Programmes through the use of empirical data and the use of evaluation frameworks are selected. Neches et al. (1991) presented a vision of the future in which knowledge-based system development and operation is facilitated by infrastructure and technology for knowledge sharing. Within and beyond the organisation’s boundaries the contemporary employee will increasingly use social computing for knowledge sharing.

Empowering all the stakeholders to engage in externalising and sharing data, information and knowledge results in a learning organisation. Progressing from data, to information answers the fundamental questions of “who", "what", "where", and "when". Going from information to knowledge we need to be able to answer the “how” question whilst understanding requires an appreciation of the “why”. Finally, wisdom is evaluated understanding (Ackoff, 1989).

Ackoff indicates that the first four categories relate to the past; they deal with what has been or what is known. Only the fifth category, wisdom, deals with the future because it incorporates vision and design. With wisdom, people can create the future rather than just grasp the present and past. *“But achieving wisdom isn't easy; people must move successively through the other categories.”* [http://www.systems-thinking.org/dikw/dikw.htm – last accessed 01/15/07].

5 *Model Credibility*

The credibility of the proposed model was establish by a verification and validation process. The verification of the KASL II model was carried out using an interpretive research method whereby five experts were interviewed by the authors. At the beginning of each interview an explanation of each component depicted by the model including all four learning loops and the various feedback loops was given to the interviewee. According to Macal (2005) “Model verification attempts to establish whether the model implements the assumptions correctly i.e. verification addresses the following questions:

(i) Does the model solve an important problem?
(ii) Does the model contain errors, oversights, or bugs?
(iii) Does the model meet a specified set of requirements?
(iv) Does the model perform as intended?”

The grading scale given was S = Strongly Agree, A = Agree, D = Disagree, SD = Strongly Disagree. The experts’ responses are shown in Table 1.

Table 1: Model Verification

Q (i)	Q(ii)	Q(iii)	Q(iv)

Expert	S	A	D	SD	S	A	D	SD	S	A	D	[redacted]	S	A	D	SD
A	√							√		√				√		
B		√					√			√				√		
C	√							√	√				√			
D	√							√	√							√
E		√						√	√							√

The responses to Questions (i) and (iii) were either an Agreement or Strong Agreement which was encouraging. The responses to Q(ii) were emphatically Strongly Disagree or Disagree which means that there were no errors, oversights or bugs. There followed a round table discussion of the experts and the researchers. The main issues discussed were the variation of responses to Question (iv) i.e. Does the model perform as intended?

The essence of the discussion is summarised in the following comments:

> *The model seems workable.*
>
> *The processes depicted make sense.*
>
> *The question should have been "Do you expect the model to perform as intended?"*
>
> *Basically performance can only be judged / measured after use, indeed after repeated use, hence we are undecided*.
>
> *It depends on what you mean by performance* – time will be the judge of this.
>
> *The efficiencies of the model will become evident with use.*

Model validation depends on the purpose of the model and its intended use. It can be considered as an exercise in "thought space" to gain insights into key variables and their causes and effects (Macal, 2005). Model validation attempts to establish whether the assumptions that were made are reasonable to the real world. We need to address the following questions:

(i) Can we ensure that the model meets its intended requirements in terms of the methods employed and the results obtained?

(ii) Is the model useful i.e. does the model address the right problem and does it provide accurate information?

The KASL II model was validated by applying it to a case study "Managing a franchise partnership" (Middlesex University, London, UK). Each partnership is managed according to the University's Quality Assurance procedures and to the specific terms agreed at validation of the link and the programme(s) (e.g. University of Nicosia, Cyprus and Regional IT Institute, Cairo, Egypt) (http://www.mdx.ac.uk).

Individual, groups, departments (who) involved have defined roles and responsibilities (what). The methods used (how), the sequence and timing of events such as boards of study (when) as well as the location (where) are agreed and planned. In carrying out their activities all individuals and groups involved gain knowledge and experience which is explicitly documented and shared. Opportunities for reflection and evaluation (self and peer) are informed and supported by quality assurance mechanisms, reporting templates and reports, committee meetings, boards of study, examination boards, Individuals involved are learning and sharing knowledge through the use of ICTs (the internet, Virtual Learning Environment, Video-conferencing, webcams etc) which engender and facilitate the creation and progress of a Community of Practice (CoP). There remains the one question namely the ***why*** which is the reason we engage in such a provision as a School and as a University. The answer to this is encapsulated in the University's mission to service the local, national and international community with high level education provision. Students want to

Loop 1 (operational) Individuals Involved

Link Tutor (at Middlesex and at partner Institutions)

Administrator (at Middlesex and at partner Institutions)

Programme Leader (at Middlesex)

Module leader (at Middlesex)

Module Tutor (seminars/laboratory sessions at partner institution)

Technician (at Middlesex and at partner Institutions)

Learning resources officer (at Middlesex and at partner Institutions)

Student (based at partner institutions)

Loop 2 (Operational) Groups Involved

Students (cohort)

Teams (within modules mainly for group coursework activities)

Lecturers (module leaders and seminar/tutorials leaders)

Support staff (Administrators, Technicians, Librarians)

Loop 3 (Tactical) Entities involved

The Department

The School (Faculty)

The teaching team

The support team

Loop 4 (Strategic) Entities involved

The School (Faculty)

The University

Higher Education Sector

Society at large

The results of both the verification and the validation exercises gave clear indications that the model is both useful and workable.

The limitations of this investigation are due to the limited number of experts participating at the verification stage, and the fact that the model was applied to one case study. According to Galliers (1992) case studies are “an attempt attempt at describing the relationships which exist in reality, usually within a single organisation or organisational grouping”. Although case studies is by far the most frequently used reserach method which captures reality, generalisability is difficult due to the problems relating to acquiring the same or similar data from a statistically meangful number of cases. In this type of studies a degree of subjectivity creeps in die to different interpretations of events by individual researchers and respondents.

6 Conclusion and Future Perspectives

In this paper we provided background for supporting the view that in workplaces in general but particularly in virtual workplaces there is a need for a methodology for identifying and converting individual expertise, skills, and experience into organisational knowledge that is strategically aligned with organisational knowledge transfer and learning investment into organisational value outcome. We proposed the dynamic four loop model for knowledge sharing and learning (KASL II) for aiding the knowledge sharing process and hence learning through showing the stages of translating an organisation’s mission and goals into objectives, and how decisions and actions operate for materialising these objectives. The application of the model to everyday processes will ensure that the output of every team adheres to the company’s overall strategy.

In future, the KASL II model will be applied in industrial, training and additional educational institutions in order to obtain process improvement metrics which will in turn improve the maturity of the organisations involved.

Acknowledgements

We extend our thanks to the anonymous reviewers and their constructive comments and suggestions. We also like to thank the experts for their participation in the model verification exercise.

7 References

Ackoff R. L. (1989). From Data to Wisdom, Journal of Applies Systems Analysis, Volume 16, 1989 p 3-9.

Alavi M., Leidner D.E (2001). "Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues", *MIS Quarterly,* 25:1, pp. 107-136

Bal J.,Teo P. K. (2000). Implementing virtual teamworking. Part 1: a literature review of best practice, *Logistics Information Management*, Vol. 13, No. 6, pp. 346-352

Balstrup B. (2004). *Leading by Detached Involvement – Success factors enabling leadership of virtual teams,* MBA Dissertation, Henley Management College, UK

Barab S. A., Kling R., Gray J. H. (2004). *Designing for Virtual Communities in the Service of Learning,* Cambridge University Press, Cambridge, UK

Bettoni M., Andenmatten S., Mathieu R. (2006). Knowledge Cooperation in Online Communities: A Duality of Participation and Cultivation, in Peter Feher (ed), *Proceedings of 7th European Conference of Knowledge Management (ECKM06),* 4-5 Sept., Public Academic Conferences Ltd. Reading, UK, pp. 36-42

Davenport T. H., Prusak L. (1998). Working Knowledge: *How Organisations Manage What They Know,* Boston, Harvard Business School Press

Davis J. G., Subrahmanian E., Westerberg W. (2005). The "global" and the "local" in knowledge management, *Journal of Knowledge Management,* Vol., No. 1, pp. 101-112

De Carolis M., Corvello V. (2006). Multiple Competences in Distributed Communities of Practice: The Case of a Community of Financial Advisors, in Peter Feher (ed), *Proceedings of 7th European Conference of Knowledge Management (ECKM06),* 4-5 Sept., Public Academic Conferences Ltd. Reading, UK, pp. 116-125

Elron E., Vigoda-Gadot E. (2006). Influence and Political Processes in Cyberspace: The Case of Global Virtual Teams, *International Journal of Cross Cultural Management,* Vol. 6, No. 3, pp. 295–317

Galliers R. (ed) (1992). Information Systems Research, Issues, Methods and Practical Guidelines, Blackwell Scientific Publications

Georgiadou E., Siakas K., Berki E. (2006). Knowledge Creation and Sharing through Student-lecturer Collaborative Group Coursework in Peter Feher (ed), Proceedings of 7th European Conference of Knowledge Management (ECKM06), 4-5 Sept., Public Academic Conferences Ltd. Reading, UK, ISBN 978-1-905305-26-5 Book, pp.678-689

Gurteen D. (1999). Creating a Knowledge Sharing Culture, Knowledge Management Magazine, Volume 2, No. 5, February, Available on-line at http://www.gurteen.com/gurteen/gurteen.nsf/0/FD35AF9606901C42802567C70068CBF5/

Handy C. (2000). *Trust and the Virtual Organisation* (HBR OnPoint Enhanced Edition). HBR On Point, June 2000

Harorimana D. (2006). Knowledge Networks: A Mechanism of Creation and Transfer of Knowledge in Organisations? in Peter Feher (ed), *Proceedings of 7th European Conference of Knowledge Management (ECKM06)*, 4-5 Sept., Public Academic Conferences Ltd. Reading, UK, pp. 211-222

Holsapple C. W. (2005). The inseparability of modern knowledge management and computer-based technology, *Journal of Knowledge Management,* Volume 9, Number 1, 2005 , pp. 42-52(11)

Järvenpää S. L., Leidner, D. E. (1999). *Communication and Trust in Global Virtual Teams, Organization Science,* 10, pp. 791 – 815

Kanter V. D. (1999). Knowledge Management, Practically Speaking, *Information Systems Management*, Fall, pp. 7-15

Kautz K., Thauysen K. (2001). Knowledge, Learning and IT Support in a Small Software Company, *Journal of Knowledge Management*, 5 (4), pp. 349 -357

Kautz K., Kjaergaard A. (2008). Knowledge Sharing in Software Development, in P. A. Nielsen, K. Kautz. *Software Processes and Knowledge, Beyond Conventional Software Process Improvement,* Software Innovation Publisher, Aalborg University Denmark, pp. 43 - 68

Kwai Fun IP R. and Wagner C. (2007). Weblogging: A study of social computing and its impact on organizations, ScienceDirect – *Decision Support Systems*, Elsevier B. V. pp. 1-8

Lave J. and Wenger E. (1991). *Situated learning: Legitimate peripheral participation.* Cambridge and New York: Cambridge University Press.

Lipnack J., Stamps J. (1997). *Virtual Teams: Reaching across space, time and organisations with technology,* Wiley, New York

Macal C.M. (2005). Model Verification and Validation, Workshop on "Threat Anticipation: Social Science Methods and Models", The University of Chicago and Argonne National Laboratory, April 7=9, 2005, Chicago, IL (retrieved on 01/03/08 from http://jtac.uchicago.edu/conferences/05/resources/VandV_macal_pres.pdf)

Malhotra Y. (1996). Organizational learning and learning organizations: an overview, http://brint.com/papers/orglrng.htm,

Mansour-Cole D. (2001). Team Identity Formation in Virtual Teams. In S. Beyerlein, M. Beyerlein and D. Johnson (Eds.) *Virtual Teams, Advances in interdisciplinary studies of work teams*. Volume 8. Oxford: Elsevier Science

Mumford E. (1981). Defining systems requirements to meet business needs: a case study example, *Computer Journal*, 28, 2

Neches R., Fikes R. Finin T., Gruber T., Patil R., Senator T., Swartout W. R. (1991). Enabling technology for knowledge sharing. *AI Magazine,* 12(3): 16-36

Nonaka I. (1991). The knowledge-creating company, *Harvard Business Review,* Vol. 69 No.6, pp. 96-104

Paulzen O., Doumi M., Perc, Primoz A., Cereijo-Roibas A., (2002). A Maturity Model for Quality Improvement in Knowledge Management, *ACIS 2002 Proceedings.* Paper 5.

Siakas K., Siakas E.(2008). The need for trust relationships to enable successful virtual team collaboration and software outsourcing, *The International Journal of Technology, Policy and Management special issue on Human Aspects of Information Technology Development*, Inderscience Enterprises Ltd., Vol. 8, No.1 pp.59 – 75

Siakas K. V., Balstrup B.(2006). Software Outsourcing Quality Achieved by Global Virtual Collaboration, *Software Process: Improvement and Practice (SPIP) Journal*, John Wiley and Sons, Vol. 11, no. 3 (May-June), pp.319-328

Siakas K., Georgiadou E. (2006). Knowledge Sharing: Cultural Dynamics, in Peter Feher (ed), Proceedings of *7th European Conference of Knowledge Management (ECKM06)*, 4-5 Sept., Public Academic Conferences Ltd. Reading, UK, ISBN 978-1-905305-26-5 Book, pp. 505-513

Siakas K. V., Balstrup B. (2005). Globalisation - Sourcing by Virtual Collaboration? In Miklos Biro, Richard Messnarz (eds.) *European Software Process Improvement (EuroSPI 2005), Conference Proceedings 11/05,* 9-11 November, Budapest, Hungary, ISBN: 9638431946, Johnson Neumann Computer Society, pp. 3.11- 21

Spiegler I. (2000). Knowledge Management: A New Idea or a recycled Concept?" *Communications of the Association for Information Systems (AIS),* June, Volume 3, Article 14, pp 1-23

Storck J., Hill P. (2000). Knowledge Diffusion through 'Strategic Communities', *Sloan Management Review,* Vol. 41, No. 2, pp. 63–74

Schwen T. M., Hara N. (2003). Community of Practice. A Metaphor for Online Design? *The Information Society,* 19, pp. 257-270.

Tanriverdi H. (2005). Information Technology Relatedness, Knowledge Management Capability and Performance of Multi-business Firms", *MIS Quarterly,* June, Vol. 29 No. 2, pp. 311-334

Wenger E. (1999). *Communities of Practice: Learning, Meaning and Identity,* University Press, Cambridge.

Wenger E., Snyder B. (2000). Communities of Practice: The Organisational Frontier, Harvard Business *Review, Vol. 78, No.1, 139–145*

CMMI based Continuous Improvement Program in a financial software company – Results and Experiences

Erkki Savioja, Tieto Oy/ Fidenta, Aleksis Kiven katu 3-5, 00530 Helsinki, Finland
erkki.savioja@tieto.com

Risto Nevalainen, FiSMA ry, Vaisalantie 4, 02130 Espoo, Finland
risto.nevalainen@fisma.fi

Abstract

Software process improvement is a specific topic of management. It is compared often to change management and/or quality management due its many similarities with them. But SPI has also its own disciplines and best practices. One basic strategy for SPI is based on systematic use of models. Models can be either assessment or improvement oriented, like well known CMMI and SPICE.

This paper presents a case where the motivation to use a maturity model (CMMI) is clearly on an improvement territory. The results of annual appraisals in a medium size Finnish software company are presented. Also the dependency between appraisal results and improvement efforts done during the previous year are reviewed. Results are presented both as a capability/maturity level and a capability index for each assessed process. Capability index is developed as an additional presentation to standard process capability scale, to verify the soundness of the capability and maturity level result and to show more refined and detailed results.

Relationships between model based results and other business results are also analysed and presented. Fidenta has collected systematically measurement data since 1997. In this article we make trend analyses from data during years 2005 – 2008. This case article shows positive results from systematic use of models. It also shows that SPI is long-term action, and expected business results are quite difficult to verify in a quantitative way.

1 Introduction

During last years the business concepts within the software industry has been under rapid change. The pressure to higher quality by lower expenses e.g. through distributed software development has been extensive. Mature and consistent processes as well as a right alignment of SPI (software process improvement) initiatives are precondition for the success in this kind of more and more complicated operational environment.

Key targets of SPI activities are to support company to achieve its business goals and keep the customer satisfaction on a high level. Improved maturity and capability of software-processes are seen as a procedure to decrease quality creeping of delivered products. [1]

The case study presented in this paper is a maturity model (CMMI) based SPI program run in a middle size Finnish software company, specialized in financial software development.

Results of B-type assessments in 2005-2008 are described both through achieved capability levels and FiSMA process capability index (PCI). 13 process areas have been included in this case study material. Assessment results are presented as averages and trends at aggregate level. Risk management process area is represented in more detail.

This article presents analysis and trends of results and evaluates advantages and drawbacks of the usage of a maturity model based SPI as follows:

- Section 2 describes background and different phases of software process improvement in the company
- Section 3 explains Tieto's and Fidenta's approach in using maturity assessments for process improvement
- Section 4 describes the assessment data and analysis of the data
- Section 5 deals with process improvement actions in 2005 - 2008
- Section 6 compares model and business results during 2005-2008
- Section 7 draws conclusions

2 Company's approach to software process improvement

2.1 About the company

This paper deals with the medium size software company in Finland. The name of the company is Fidenta. It has about 220 employees and it is jointly owned by two large Nordic companies, Tieto and Nordea.

Tieto is an information technology corporation providing IT, R&D and consulting services, with over 16000 experts in more than 25 countries. It is among the leading IT service companies in Northern Europe. Nordea is one of the biggest banks in the Nordic countries and Baltic area. Nordea has been established through many merges and acquisitions and it started to operate with the current name in 2001.

The structure of Fidenta's partnership model is described in the figure 1.

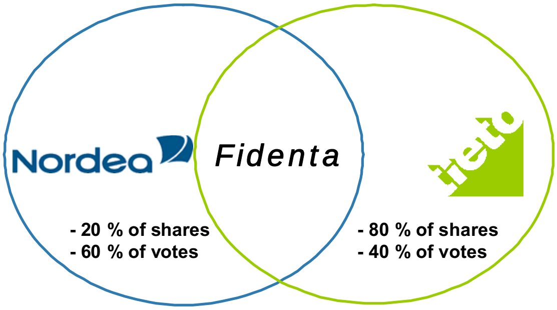


Figure 1. Joint venture structure of Fidenta

The company structure of Fidenta is built on a close partnership with the customer. Owner companies have their representatives on the board of Fidenta, Nordea (the customer) has the majority of the votes and Tieto has the majority of the shares. The fundamental idea behind the partnership is that the customer can concentrate on banking business while Fidenta provides IT expertise and takes care of developing and maintaining IT systems of banking applications benefiting Tieto's wide competence network when necessary.

The efficiency and win-win situation are based on trust, high transparency, customer centricity and a long term close cooperation where Fidenta is familiar with customer's procedures, methods, terminology, organization, regulations and people.

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

2.2 Background of quality improvement efforts

Fidenta has done systematic software process improvement since it was founded in 1994. High quality was identified as a competitive edge of the company from the beginning. Best practice and experience sharing within the Tieto group has been part of Fidenta's quality work throughout the years.

The development of quality thinking in Fidenta is outlined through following phases:

- ISO 9000 Certification 1995 – 1997
    - The decision to attempt to attain a quality certificate was made and ISO9000:2000 certificate was achieved in 1997
    - Focus areas: Company culture, Teamwork, Quality
- Development of Processes and BSC Measurement 1998 – 2000
    - The process based business model was established with implementation of balance score card (BSC) related measurement practices; quality handbook was replaced by process descriptions including quality requirements
    - Focus areas: Project work process, BSC Measurement, Business Model
- Networking with Nordea and Tieto 2001 – 2003
    - During this phase annual follow-up audits of ISO9000 quality certificate were used for identifying process improvement needs and for monitoring of efficiency of improvement efforts made
    - Focus areas: Going International, Networks, Customer Focus
- Right Competence – High Performance 2004 – 2006
    - In 2004 the management of Tieto group launched CMMI as a corporate level assessment model, since then Fidenta has run combined CMMI B-type and ISO 9000 assessments
    - Capability and maturity model has replaced the ISO follow-up audits as a primary method to initiate SPI needs
    - Focus areas: Nordic co-operation, Nordic partnership
- Presence in Northern Europe 2007 – 2009
    - It was seen that company's commitment to the quality was difficult to communicate to the customer by a combination of capability levels of certain processes. The maturity level 3 was set as a target because a maturity level is simpler and easier to understand for stakeholders
    - Focus area: Maturity of project delivery and application service processes

3 Maturity model usage since 2004 in Fidenta

3.1 Annual B-type assessment according to Tieto policy

Fidenta got the ISO9000:2000 quality certificate in 1997. It was upgraded to ISO9001:2000 in 2003. Until 2003 Fidenta's performance was audited against ISO9001 requirements and process improvement initiatives were mainly build up of the results of annual follow-up audits of the certificate.

Tieto made a corporate wide decision in 2004 that all units should assess their performance against CMMI model requirements. [2] Since that Fidenta has carried out combined CMMI B-type and ISO9001 assessments on annual basis. In the assessor team there has been three external assessors and three internal Tieto-CMMI Lead assessors.

In 2004 Tieto started also a training program for internal CMMI assessors and lead assessors. In Fidenta about ten people have passed the assessor exam and three the lead assessor exam.

A corporate level knowledge sharing of CMMI experiences has taken place in annually arranged Tieto-CMMI Days. More than twenty units have been represented in the event and Fidenta gave a presentation of its approach and results in 2006.

3.2 Target setting linked to the business goals

Target setting is based on the analysis of current situation and strategic business goals and it is done by the Fidenta Management team.

Besides maturity and capability related targets Fidenta has always stated also targets concerning actual improvement needs from business and management perspective. Assessors are expected to take those business goals related issues into account in their assessment report. All improvement proposals of assessors are linked also to the business goals.

Fidenta has each year stated also one or two non-model questions to be examined in the interviews.

In the first place assessments were totally related to identify process improvement needs. In 2007 management stated that the maturity level three would be targeted. Due to increased awareness of customers for the CMMI and maturity levels the management wanted to take also the maturity dimension in use in assessments.

3.3 Assessment results as input to the annual software process improvement program

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

The focus in choosing development efforts has been on how effectively they enhance achieving of company's business goals. Model requirements have been regarded as secondary reasons only. According to the development roadmap Fidenta is aiming to get its project delivery and application service management process on level three. The roadmap is based on Fidenta's strategy plan and the analysis of assessment results.

Implementation is done through internal development projects, which are run according to the same project delivery process as customer projects with planning and control elements. The crucial part of implementation phase is that results are communicated to the staff so that all employees become familiar with the effects of improvements to the every day life in the company. Internal development projects are set and results are accepted by the Management team.

Results are monitored in biannual quality reviews by the Management team, but the main forum to evaluate the results of improvement efforts is next CMMI assessment. Improvement proposals and possibilities mentioned in the assessment report are again an input to the target setting for the next year's improvement steps.

4 Assessment data and analysis

4.1 Data collection

Fidenta has done various kinds of assessment for a long time. In this presentation we cover and analyse only annual CMMI assessment results and trends during years 2005 – 2008. This data is quite homogenous and reliable, because same assessment team has done all assessments. Also assessment scope has been almost the same. We cover 13 process areas, which have been assessed in all four years. In last two years all 18 process areas included in maturity level 3 have been assessed. All assessments are type B, as defined originally in Scampi model and adapted for Tieto Oyj. Tieto has used CMMI systematically from 2004 and performed at least 100 B-type assessments since then.

CMMI data and results include evidences, their mapping with model elements, ratings of each model element at capability levels 0 – 3 and aggregated results by capability and maturity level. Each assessment had 4 – 6 instances, 300 – 600 documents and 25 – 30 interviews. In chapter 4 we analyse mainly CMMI results only.

Fidenta collects systematically also process performance data, like customer satisfaction, employee satisfaction, financial data and investments in knowledge. This data was available for us as evidence during each assessment. In this article we try also to find relationships with CMMI data and business performance data during years 2005 2008 (see chapter 5).

4.2 Process capability index

CMMI results in Fidenta consist from detailed ratings of each required and expected model element (specific practices, generic practices, specific goals, generic goals, capability levels). Most important results are detailed ratings at practice level. We used 4-point scale NPLF as rating scale for all practices and goals. In this analysis we use only consolidated results.

CMMI results were also converted to metric, what we call process capability index (PCI). It was originally defined by FiSMA, and is an essential part of Tieto Oyj adaptation of CMMI. Each rating was changed into numeric value between 0 – 1. These values can be used to calculate averages, sums etc. Each practice rating in CMMI is assigned to one level at one process area. Similarly also PCI is a numerical value for each level of any process area.

Table 1. Numerical values of ratings for process capability index

Rating value	Numerical value for index element (specific practice, generic practice)
NR	--
NA	--
N	0
P	0,33
L	0,67
F	1,00

Scale of PCI is continuous and gives better comparability between assessment results than just CMMI ratings. Also various new PCI based presentations were used, like “traffic lights” and “benchmarking presentation”. Because PCI value has the same origin as CMMI rating, it correlates strongly with that and can be seen as one additional presentation of assessment results.

Process capability index is also comparable between different models, if their most detailed elements can be mutually mapped. For example, CMMI and SPICE results can be compared and even unified with PCI result, even their logic is quite different in process capability rating. That unification can be done, if all levels 1 – 3 are in calculation. CMMI and SPICE detailed elements, especially generic practices at CL2 and CL3, are not fully comparable inside one capability level.

4.3 Process capability level and index results and trends 2005-2008

We can publish only average results and trends here, because full results are confidential. Table 2 contains average results from 13 processes. Trend in capability level (CL) data is increasing quite systematically. This result indicates that Fidenta is approaching capability level 3 of all selected processes. When capability level is achieved by all process areas, average results will be 3.0. If all 18 process areas of CMMI ML3 are at CL3, also maturity level 3 is achieved.

Table 2 Average result of capability level (CL) and process capability index (PCI) during 2005 – 2008

Year:	2005	2006	2007	2008
Average value of CL:	**2,09**	**2,38**	**2,70**	**2,69**
Average value of PCI	**2,42**	**2,48**	**2,66**	**2,43**

Capability index result shows a bit different trend than CMMI capability levels. It increased for first three years, but then declined from value 2,70 to value 2,43. There are several reasons for this:

First of all, PCI measures a bit different things than capability level. PCI is a kind of “conformance measure”, showing degree of achievement of capability at selected levels. In ultimate case up to capability level 3, PCI value would be also 3.0. It means that all detailed ratings would have value “F”, Fully Achieved. It happens almost never in real assessments. Each “L” rating drops PCI value a bit, even it is not seen in capability level result.

Even several “L” ratings at capability level 1 (specific practices) do not drop CL result, if they are not classified as a whole as major weakness according to Scampi. But that should be shown as “P” or “N” rating at specific goal level. Normal case is that L-ratings lead to L-rating of that specific goal, according to “weakest link” principle of CMMI. So, in most cases capability level result is better than capability index result when all levels 0 – 3 are included.

It can happen also that some specific practice gets P-rating and then CL result will be 0 (zero). Anyway, PCI result can be quite high, showing compensation by other ratings.

In Fidenta case, both results were used. Capability level result was used to check gap between target level and achieved level. PCI result was used to compare results between other units in Tieto, and with previous results. So, it was used like a benchmark.

PCI data allows various other kinds of analyses. One example is average PCI result by each capability level, see Figure 2. It tells quite clearly, what is the level of “institutionalisation” of each process area in Fidenta during 2005 – 2008. Of course, same data could be shown as trend diagram etc.

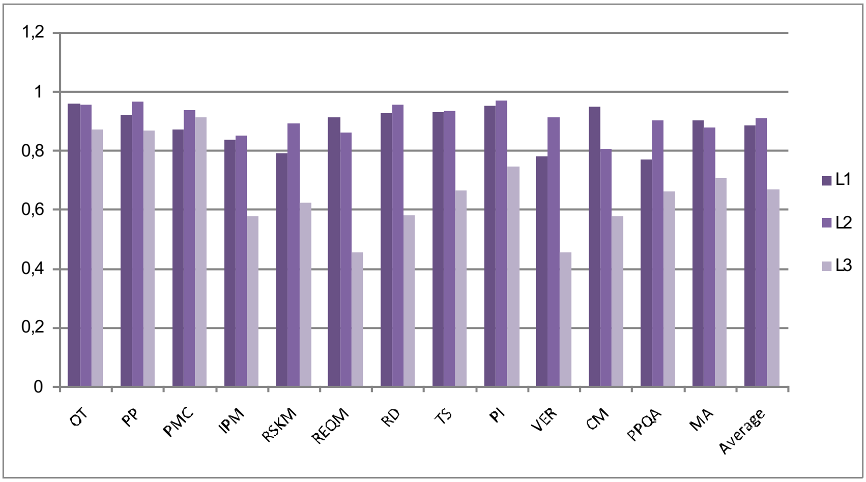


Figure 2. Average PCI values by level of each process area during 2005 – 2008

4.4 Risk management process area as an example

We selected risk management (RSKM) as one example to show process area specific results and trends in Fidenta. Result is presented in table 3.

Table 3. Process capability level and PCI results from risk management process area

Year:	2005	2006	2007	2008
CL result of RSKM:	**0**	**3**	**3**	**3**
PCI result of RSKM:	**1,83**	**2,54**	**2,58**	**2,31**

Results show clear differences between CL and PCI results. In 2005, achieved CL is low even PCI is rather good. In 2008, result is opposite showing good results for CL but only moderate result for PCI. We can make three additional conclusions from that, based on Fidenta experiences:

CL result has large variation, being anything between 0 – 3. It is very difficult to see any trends from those results. Main reason is the “weakest link” principle, leading to one “P” rating in 2005 and then to low CL result. In other years, situation is opposite. Even several “L” ratings allow CL value 3 as result. So, CMMI result can be either too low or too generous, depending on evidences and selected instances.

PCI results show clear trend, which is increasing during 2005 – 2007, but declines in 2008. There are several explanations for this. Maybe most important are the selected instances (projects), which are different each year. So, there can be random variation in PCI results, as well as in CMMI results.

PCI does not measure quite properly at capability level 3, because CMMI model has only two practices at CL3, and then we have only two ratings as input for PCI calculation. That causes unnecessary variation, which explains most of lower value in 2008.

5 *Improvement actions during 2005-2008*

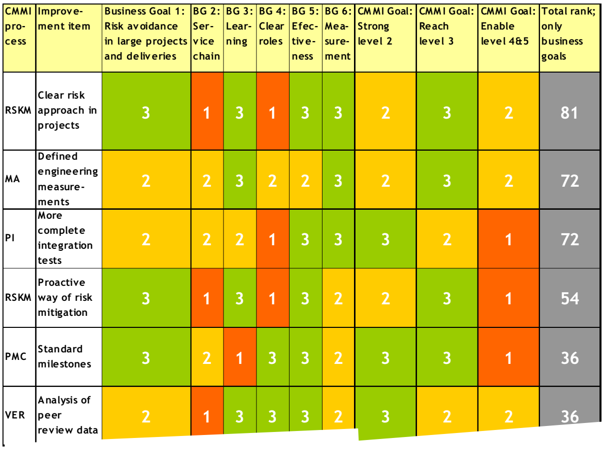
5.1 Analysis procedure

Based on a detailed assessment report, assessment team created an excel chart to analyse the findings. Idea of chart is described in the figure 3. Each model requirement where the rating was not "fully" achieved is listed in the chart. These are all considered as potential improvement initiatives. The reason, what was missing, is also described in the chart.

All the items have then been valuated against the business goals and the model goals related to different levels of the model with a simple 1-3 scoring, where three means that an improvement of this specific item, would have high influence to achievement of the goal. Value 1 means that the influence is low or no influence at all. This has been done for all business goals and for maturity levels 1, 2, 3 and 4&5 of the model.

Valuation enabled to rank improvement initiatives according to each column (business and model goals) and get a view what are the most effective actions to improve different goal areas.

By multiplying the valuations of a single improvement initiative it has generated combined values for business goals only, for model goals only and for total priority covering both business and model goals. The tasks of the annual process improvement projects have been selected based on this analysis.



CMMI process	Improvement item	Business Goal 1: Risk avoidance in large projects and deliveries	BG 2: Service chain	BG 3: Learning	BG 4: Clear roles	BG 5: Efectiveness	BG 6: Measurement	CMMI Goal: Strong level 2	CMMI Goal: Reach level 3	CMMI Goal: Enable level 4&5	Total rank; only business goals
RSKM	Clear risk approach in projects	3	1	3	1	3	3	2	3	2	81
MA	Defined engineering measurements	2	2	3	2	2	3	2	3	2	72
PI	More complete integration tests	2	2	2	1	3	3	3	2	1	72
RSKM	Proactive way of risk mitigation	3	1	3	1	3	2	2	3	1	54
PMC	Standard milestones	3	2	1	3	3	2	3	3	1	36
VER	Analysis of peer review data	2	1	3	3	3	2	3	2	2	36

Figure 3. Valuation chart of improvement initiatives identified in the assessment

5.2 Focus areas of annual process improvement projects during 2005 – 2008

Fidenta has established each year an internal development project for process improvement. The tasks of the projects have been generated from different sources e.g. collected improvement initiatives from employees and management, analysis of projects' initial and technical review data, measures and EFQM self assessment results. But the results of CMMI assessments have been always the most important source of the improvement tasks.

Here beneath are described some headers of the improvement areas of annual process development projects (most relevant CMMI process area(s) also mentioned):

- Based on the results of the assessment 2005:
- Clarification of the project owner role (PP, PMC)
- Definition of new type of services and processes (all process areas)
- More systematic approach for learning and experience sharing cycle (all process areas)
- More solid and consistent risk management (RSKM)
- Risk policy (RSKM)
- More active role of the project owner in risk management (RSKM)
- Project management tool for project managers (PP, PMC)
- Based on the results of the assessment 2006:
- Technical reviews during the project (VER)
- More consistent product integration and configuration practises concerning all platforms (CM, PI)
- Development of measurement to indicate progress and quality of projects and results (MA)
- Enhanced monitoring of earned value of projects (IPM)
- A tool for company level recourse management (PMC, IPM)
- Reviews after the project and more systematic way for collecting and sharing experience data of risk management and testing

In years 2007 and 2008 assessment covered also some level 4 and 5 requirements of CMMI. Organisational training (OT) was selected as the first process area which was assessed up to level 5. Ensuring right competences of the personnel was identified as one critical success factor of the company. Good results of previous years were the reasons behind selecting OT for experimenting of higher levels.

Some other improvement topics based on the results of the assessment 2007:

- Strengthening of code and architecture reviews and sharing the findings through common intranet (VER)
- More detailed descriptions of the Formal Decision Making process (DAR) and the Distributed Software Development process (SAM)
- Better involvement of staff with the improved processes. This was done through Quality Road Shows covering all the units in the company

5.3 Further need for improvement

Again, based on the results of the assessment 2008, it is clear that there are lots of challenges in strengthening a maturity of the company and capabilities of the processes.

Since 2004, when we started to monitor the maturity and capability of our processes through the CMMI model, we have got forward on many areas and we are more aware of the practices which influence to the efficiency of our performance. We have not only achieved higher customer satisfaction, but also succeeded to increase employee satisfaction by paying attention to improving the less capable practices reviled by the CMMI assessments.

We had difficulties for a long time with certain CL 1 requirements, but now we have managed to get rid of those. Now our focus is more on process tailoring, systematic measuring, experience sharing and learning practices.

During the past 12 months we have invested a lot to social media. It is expected to boost knowledge and experience sharing as well as information sharing e.g. about new guidelines and process improvements.

Currently ongoing process improvement project based on results of 2008 assessment deals with following development areas:

- Measurement repository to serve all projects, targeting to improve estimation and learning (IPM, OPD)
- Project audits (PMC)
- During a project aiming to enable effective corrective actions while the project is still going on
- Post-mortem analysis of each project to enable learning
- Standardised testing process for projects and application services aiming to increased quality of deliveries and learning possibilities (VER, OPD)
- Systematic experience collection to a company level project asset and measurement library aiming to have one common way to run projects (IPM, OPF, OPD)
- Defined tailoring options for project planning trying to reply to the question “What planning mechanism should be applied in my project?” To get tailoring more sound there is a need to get created a project start-up kit type support tool for project managers (PP, IPM)

6 Model results vs business results during 2005 – 2008

Fidenta has collected systematically measurement data since 1997. More than 50 metrics is standardised. Data collection is mainly supported by tools, so that data is also quite reliable and covers all instances. Most of metrics has also remained same since 1997, enabling trend analyses and comparisons. Focus has been mainly in customer satisfaction, financial results and project specific metrics.

Table 4 presents a small sample of business related measurement data. Two last rows are the same as in chapter 4 table 2, to allow qualitative analysis.

Table 4. Some business measures vs CMMI related results

Year:	2005	2006	2007	2008
Customer benefit index 1-5:	4,1	4,2	4,2	4,2
Project feedback index 1-5:	4,4	4,3	4,3	4,2
Employee satisfaction survey:	566	595	635	678
Growth of invoicing per capita, %:	-4,8	2,5	13,0	4,7
Invoicing rate, %	61,9	66	67,1	65
Investment in training per capita, k€:	1,6	2,2	2	2,1
Head count, total:	209	212	208	210
Average value of CL:	2,09	2,38	2,7	2,69
Average value of PCI:	2,42	2,48	2,66	2,43

Basic quantitative analysis about mutual relationships of data was done by correlation calculation. It shows similar type of behaviour between data sets, but does not necessarily explain real causal relationships. Some selected areas of strong correlation are (potential explanations are also presented):

- Employee satisfaction vs. achieved capability level. This could be explained by “better process, better results” combination, which is of course very motivating and stimulating also for staff.
- Also correlations between CMMI level index and customer benefit and business growth were relatively high. Is it so, that when business is growing, it causes better customer satisfaction?

As interesting as high correlations, are also low correlations. It shows that data sets seem to have no clear common behaviour. Most of correlations are relatively weak. It shows that each measurement is an indication result of its own. For example, correlation between main business results and capability level index is systematically quite low.

7 Conclusions

In this paper, we have described a maturity based SPI program run systematically for five years in a medium size Finnish software company.

It has been pointed out often that management commitment is most important single issue to secure expected results and return of investment of a process improvement program. In Fidenta case management has stood strongly behind the process improvement initiatives and it has without question helped to succeed in SPI efforts of the company.

In spite of severe intentions targets may remain unreached due to too short term benefit expectations. Company wide changes in people’s behaviour require systematic, long term efforts and take its time. It is unrealistic to expect immediate results. One of the reasons for failures could also be the different opinions about the focused goals.

Important dimension in using maturity models is to keep in mind clearly that all SPI efforts should support achievement of business goals and not only to fulfil the requirement of the model.

One disadvantage in the CMMI model is its complexity. It is challenging to understand the link between company’s own, more generic processes and the process areas of the model. E.g. Tieto has one harmonised corporate level project delivery process which covers many process areas of the model.

Another inconvenience related to the complexity of the model is how to communicate the model, its requirements and results of the assessment in understandable way for the people who are not familiar with the model at all.

It is also a challenge to keep the trend to higher capability index values in a situation when working procedures change rapidly. Another challenge is how to keep up the motivated atmosphere, when the progress is not always so clear and fast. It could also question how the criteria of maturity model succeed to follow changing every day life processes and practices in software industry companies.

Another important question for the future is financial dimension, how to get ROI and competition benefits out of process maturity assessments in most effective way.

8 References

1 Erkki Savioja, Markku Tukiainen: Measurement practices in financial software industry. Software Process Improvement and Practice 12(6): 585-595 (2007)

2 TE CMMI policy, Operational and Development Management Team of TietoEnator, May 2004

3 FiSMA PCI calculation rules. Presented in CMMI training materials for Tieto.

4 CMMI 1.1, 1.2. Software Engineering Institute.

5 Scampi 1.1, 1.2. Software Engineering Institute.

9 Author's CV

Erkki Savioja has over 20 years IT experience of financial sector in quality assurance and project management roles. Measurement and evaluation methods, project and risk management and software process improvement are his interest areas. He has been Fidenta's representative since 1998 in the knowledge sharing network Finnish Software Measurement Association (FiSMA), chair in 2006-2007. He has graduated from Turku University with computer science as a main subject.

Risto Nevalainen (Lic.Techn.), Senior Advisor in FiSMA. Risto Nevalainen has long experience in software measurement and quality topics. His working experience includes position as managing director of Finnish Information Technology Development Center during 1989-1995. Mr. Nevalainen has participated in ISO15504 (SPICE) standard development since beginning. He is Competent SPICE Assessor and ISO9000 Lead Assessor.

Lessons learned from an ISO/IEC 15504 SPI Programme in a Company

Antonia Mas, Bartomeu Fluxà, Esperança Amengual

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

Brújula Tecnologías de la Información S.A
Edificio U-Parc BIT Ctra. Palma-Valldemossa, Km 7,4 - Palma de Mallorca, Spain
tfluxa@brujula.es

Abstract. This paper describes a first hand experience in Software Process Improvement within a Spanish company which, since 2002, has been involved in an improvement program led by our research group. We discuss the experience and the results of this improvement programme, as well as the lessons learned to deal with new future improvement initiatives in other companies of our environment.

1 Introduction

One of the main interests of our research group is to promote software process improvement within the companies of our environment. We think that experimentation in research is crucial since it allows demonstrating the applicability of the proposed solutions.

With this interest in mind, in 2002 we led a SPI project in eight small companies in the Balearic Islands. The QuaSAR project [1] was an initiative that allowed us to analyse the software development sector in our autonomous region, and it provided guidance to the companies for the improvement of their software processes and for the ISO 9001:2000 certification. The QuaSAR project was a successful initiative which motivated the companies for continuous process improvement.

Nowadays some of the QuaSAR participants still continue working for the improvement of their software processes. In this article we expose the experiences and the results of a SPI programme in one of these companies. This improvement programme has been placed into a grouped action with other six companies with a dual objective. On the one hand, the main goal of the improvement initiative is the identification of synergies among participants to share and spread the knowledge and project experiences. This is an ambitious aspect of the programme since it implies a cultural change in a group of small enterprises towards a collaborative association, further than the individual benefits of each company, which has induced to the formation of a group with a common objective: the technological excellence. On the other hand, the second goal is the dissemination of the improvement programme results as a public awareness and incentive for other companies and institutions to motivate them to form groups in future software process improvement initiatives.

2 Company Background

Brújula is a Spanish company which began its activity in 2000. Nowadays, Brújula has 125 employees dedicated to the development of Internet-based applications and the implementation of the infrastructure which supports them.

The implementation of a quality management system and the ISO 9000 certification obtained by the company in 2002 initiated what has become one of the main identity insignias of Brújula as a company: the quality as a management strategy. In 2005 the company introduced the EFQM Excellence Model to its management system and at the end of 2007 began the adaptation of its processes to the ISO/IEC 15504 international standard for software process assessment and improvement.

Brújula's vision is to be a large company, recognized at a national level for providing integral solutions to its clients through the continuous improvement of the quality of its products. For the company, quality is the manner of achieving its business goals, satisfying the requirements and expectations of its clients.

3 Steps of Process Improvement

The process improvement programme has followed the steps of process improvement utilizing a conformant process assessment according to the requirements of ISO/IEC 15504 described in ISO/IEC 155045-2 [2] and in ISO/IEC 15504-3[3]. In this section, the application of these steps, as detailed in ISO/IEC 15504-4[4], is described.

3.1 Initiating process improvement

From an analysis of the company's business goals and the existing stimulus to improvement, the following improvement objectives were set:

- Achievement of quality and productivity standards to allow the company technological innovation in order to gain visibility and recognition to facilitate collaboration with partners and also participation in large projects.
- Provision of high value to the clients through project management, incidence management and release of defect-free software.
- Initiation of a changing process aimed to improve the position of the company both at national and European IT sectors.

To address these business goals, the Process Reference Model provided in ISO/IEC 15504-5[5] was selected as a framework for the improvement programme. This model was the basis for the choice of the processes to be assessed and the setting of the improvement targets. Moreover, the processes referenced in ISO/IEC 15504-7[6] (Assessment of Organisational Maturity) were also considered in order to select the necessary processes to reach a maturity level 2.

As recommended by the standard, the process improvement programme was implemented as a project in its own right with defined project management, budget, milestones and accountability. Therefore, a Process Improvement Plan was produced.

The following sections describe each one of the tasks performed through the improvement programme.

3.2 Assessing current capability

In any improvement programme an initial process assessment is undertaken to understand the capability of the processes which have been previously selected for the improvement.

This assessment was conducted by an independent assessment team of two competent assessors. A result of this assessment was the identification of a set of strengths and weaknesses that were used as a basis for the development of an action plan.

The identified strengths were the following:

- The availability of a quality report which considers different organizational and management aspects, as well as some product development aspects.
- The performance of project management, problem resolution and non-conformity management activities with the support of a CASE tool.
- The identification and assignment of human resources to the different project processes based on established profiles.
- The professionalism of the employees, the individual experience of the people in previous projects, the positive work atmosphere and the fluent communication among team members.

The list of improvement actions compiled from the identified weaknesses was the following:

- Tailoring the company's standard processes to each particular project.
- Individual management of project processes.
- Explicit adjustment of plans when deviations occur.
- Management of process work products.
- Management of data about the implementation of the standard processes to demonstrate their suitability and effectiveness, as well as to identify a continuous improvement of these standard processes.

3.3 Planning improvements

3.3.1 Participants

For the implementation of the improvement project, the following roles were defined:

- Sponsor. The sponsor had the responsibility and the authority to make sure that adequate resource and competencies were made available. This role was performed by the Service and Project Manager of the company.
- Project Manager. The project manager should manage the project, validate achieved improvements, coordinate and control current projects. This role was performed by the Quality Manager.
- Organizer. The purpose of the organizer was to simplify the work of the teams and identify common objectives. Tasks of the organizer were the organization of work sessions, supporting process improvement and controlling projects. This role was performed by an assessor with experience in teamwork.
- Quality specialist. The quality specialist gave support to the definition of the processes and participated in the inspection and control of the currents projects.

3.3.2 Work groups

For the improvement of the different identified improvement areas, five work groups were formed. Each work group was responsible for a set of processes. The leader of each work group had to develop the plan and assign tasks to reach the identified objectives. Each group was formed by people with different skills and abilities to assure that they would have distinct points of view of the processes to improve. All of the members of the development area were part of one of the five groups. The following table shows these work groups.

Table 1. Work groups and improvement areas

Work group	Processes
Management	MAN.3 Project management MAN.5 Risk management SUP.4 Joint review SUP.9 Problem resolution management ENG.12 Software and system maintenance
Analysis	ENG.1 Requirements elicitation ENG.4 Software requirements analysis ENG.8. Software testing
System	ENG.2 System requirements analysis ENG.3 System architectural design ENG.9 System integration ENG.10 System testing
Design-Construction-Integration-Installation-Release	ENG.5 Software design ENG.6 Software construction ENG.7 Software integration ENG.11 Software installation SPL.2 Product release
Quality Assurance	SUP.1 Quality assurance SUP.2 Verification

	SUP.7 Documentation SUP.8 Configuration management SUP.10 Change request management

3.3.3 Action plan

The quality manager and the assessor analyzed the strengths and weaknesses identified in the previous assessment to identify: strong processes, improvement areas, and evidences of corrective actions. It was decided that the actions to be performed should obtain the planned results in nine months.

Each work group had full autonomy to approach the improvement of the processes. Different communication channels were established in order to facilitate information interchange among groups. Firstly, the intranet of the company would be used to spread the implemented improvements and to inform about their achievement level. Secondly, the corporative blog would be used to inform about technical improvements. Finally, workshops would be delivered every fifteen days.

The organizer had to give support to the different work groups by focusing the work to the achievement of the result through the definition of the processes, their documentation, the management of the tasks and the adaptation of the processes to the standard.

The quality manager had to review the results of each work group to validate if the standard requirements were satisfied. If so, the process could be published and made accessible to the organization. Otherwise, the deficiency in the process could be identified and the process would be adjusted to be adapted to the standard.

3.4 Implementing improvement actions

The implementation of the improvement actions in every project in the company was performed through joint reviews, quality controls and training actions.

The quality team coordinated and participated in periodical joint reviews with the project managers and the different specialists assigned to the projects. The main goal of these meetings was to communicate and support the understanding of the performed improvement actions. During these reviews the work products of the different processes were adjusted. It is important to notice that some of the improvements provoked and important change in the organization processes and also in the manner of the employees approached their tasks.

In parallel with the joint reviews, the quality team performed quality controls by randomly inspecting current projects, examining in each control a previously established set of processes. As a result of these controls, preventive and corrective actions where defined both at a project level and at a process level.

Throughout the project, training sessions where prepared and executed. Some of these sessions had a general interest, for instance the session to introduce the ISO/IEC 15504 standard. Other more specific sessions focussed in particular processes.

3.4.1 Internal workshops

During the first months of the project, workshops were delivered every fifteen days. The assistants to these workshops were the project sponsor, the leader of each work group, and the whole quality team: the manager, the assessor, and the specialist.

The objectives of the workshops were:

- The revision of global achievements and team progression.
- The alignment of the effort of the different work groups.
- The identification of transversal improvement areas and the decision of the optimal solution.
- The information about decisions that could affect the project advancement.

The order of the day was opened one week before the workshop allowing the inclusion of the different points to consider by any one of the assistants. During the workshop the points before established were addressed in rigorous inscription order.

As the project was progressing, the workshops were not delivered periodically. Instead, they were summoned by any one of the interested parties.

3.4.2 External checkpoints

The improvement plan included two assessments, an initial diagnosis assessment and a final certification assessment. Moreover, three external monitoring checkpoints were planned. These checkpoints allowed evaluating the level of achievement of the implemented improvements and controlling if the improved processes were aligned with the standard best practices.

From the results of these checkpoints the performed changes were validated and adjusted, as well as new corrective actions were proposed.

3.4.3 Working day with the client

With the intention of aligning the improvement actions, in the middle of the project it was decided to organize a working day with the client. A set of clients were selected, using criteria such as representative and experience, and they were invited to participate in a working day.

This working day was approached as a team dynamics in which the clients could present their problems, experiences, and the improvement areas that they had identified in their projects. The leaders of each team assisted to this session as responsible of the projects progress.

The analysis of the results of this session confirmed that the effort made and the approach of the improvement project were appropriate, since the improvement areas identified by the clients were aligned with the improvement actions that were being performed internally in Brújula. Moreover, from the ideas provided by the clients, it was possible to refine and adjust some of the resultant work products.

3.5 Final assessment

At the moment of planning the final assessment, it was decided to reduce from twelve to eight the number of processes to be evaluated, and plan a checkpoint for the other four processes included in the improvement project. The main reason for this decision was that some of the improvement actions to perform were still in an implementation phase in some of the projects. Moreover, they depended on new software tools that had been acquired recently at that moment. Therefore, since these improvement actions had not been performed in all of the projects, there was no sense in assessing these processes.

The result of the final assessment was satisfactory because, although the established goal was not reached, four of the eight assessed processes achieved a capability level 2. The other four processes reach a capability level 1 with a largely or fully achievement of the 95 percent of the level 2 attributes. With some more adjustments in the capability level 1 base practices for these processes, it would be possible to reach a capability level 2.

Regarding the checkpoint results, it is important to highlight that the improvement actions performed in three of the four processes achieved the necessary quality level to reach the objective. Therefore, it was confirmed that the considered improvement actions were appropriate.

4 Lessons learned

Throughout the performance of the improvement actions a set of weaknesses have directly influenced the project schedule and resources. These weak points are the following:

- Lack of knowledge of the standard. At the end of the improvement project it has confirmed that the importance of understanding the standard and interpreting its best practices for each one of the software life cycle processes was underestimated.
- Higher effort than expected. The effort for adapting the project tasks to the best practices defined by the standard was not appropriately estimated.
- Implementation of changes in the processes and information about them. Information about new established tasks, concept assimilation and the deployment of new manners of working were hindered by an aggressive planned schedule.
- Project support tools. The implementation of a set of the best practices recommended in the standard has shown that it is essential to have case tools to support the work.

On the other hand, the strengths that have been the key for the success of the improvement project are the following:

- The active participation, consciousness and motivation of all the participants in the project.
- The role of the organizer. This was a key role since it gave continuous support to each team, helping them to approach their objectives and acting as an interface to achieve coherence among common improvement areas.
- Periodical monitoring of the improvements in the current projects. The improvement programme had a very ambitious schedule and the number of improvement actions was important. Without the joint reviews and the understanding of the improvement actions by the team, the cost of the implementation of the improvement would have been higher.
- Support tools. The availability of a case tool for project management and change control make easier the achievement of some process base practices. The acquisition of a case tool as Enterprise Architect also facilitated traceability, configuration management and verifications.

5 Conclusions and Further Work

The improvement programme presented in this article has allowed the achievement of a capability level 2 in a set of processes which were considered crucial for the company.

It is not ventured to affirm that this experience has provoked a continuous improvement feeling in the company. At this moment, the head offices are not the only who promote a change, instead this change is also requested by the different departments and people in the company. However, for a successful Plan-Do-Check-Act cycle there are other two fundamental circumstances. On the one hand, the established processes need to be consolidated and implemented in all of the projects in the company, without any exception. On the other hand, the acquisition of new case tools is necessary to speed up the implementation and management of the processes.

It also important to highlight that the company considered in this article has been pioneer in the Balearic Islands in the achievement of a capability level 2 in some of the software life cycle processes. This satisfactory experience has motivated other companies from turisTEC, an association of software companies specialized in IT products and services for the tourism sector, to participate in a similar initiative. Nowadays seven companies are participating in a new software process improvement programme.

All of the companies from turisTEC consider that the implementation of a common set of best practices means competitive advantage. Companies which take part in this initiative will be individually better situated in the market, and as the number of participants increases the association will also have competitive advantages as a technological excellence reference, trademark of the Balearic Islands. Moreover, this standardization of the knowledge supports employee’s mobilization (with its positive and negative aspects) and allows the collaboration among companies in global, ambitious and competitive projects.

Acknowledgements. This research has been supported by CICYT TIN2007-67843- TIN2007-67843-C06-04 *“Modelos de simulación basados en ontologías y mejora de procesos para arquitecturas orientadas a servicios”,* SOAQSim.

References

[1] Amengual, E.; Mas, A.: "Software Process Improvement in Small Companies: An Experience". Industrial Proceedings of the European Software Process Improvement Conference, (September 2007), pp. 11.11-11.17

[2] ISO/IEC 15504-2:2003. Information technology -- Process assessment -- Part 2: Performing an assessment. International Organization for Standardization (2003)

[3] ISO/IEC 15504-3:2004. Information technology -- Process Assessment -- Part 3: Guidance on performing an assessment (2004)

[4] ISO/IEC 15504-4:2007. Information technology -- Process Assessment -- Part 4: Guidance on use for process improvement and process capability determination (2007)

[5] ISO/IEC 15504-5:2006. Information technology -- Process Assessment -- Part 5: An exemplar Process Assessment Model. International Organization for Standardization (2006)

[6] ISO/IEC 15504-7:2008. Information technology -- Process Assessment -- Part 5: Assessment of organizational maturity. International Organization for Standardization (2008)

Improving software processes using CMMI in a pragmatic way: experience at ING Luxembourg

Valérie Betry, Stéphane Cortina, Séverine Mignon,
Centre de Recherche Public Henri Tudor
29, avenue John F. Kennedy,
L-1855 Luxembourg
{valerie.betry ; stephane.cortina ; severine.mignon}@tudor.lu

Abstract

This paper presents a method of the Public Research Centre Henri Tudor combining CMMI process definitions with the ISO/IEC 15504 process assessment framework as a light way of improving software development processes. It describes the methodology, reports its usage at ING Luxembourg and presents lessons learnt and methodology improvement perspectives.

Keywords

ISO/IEC 15504, CMMI, CMMI for development, Process Assessment Model, Process Assessment, Software Process Improvement, Banking Sector.

1 Introduction

Deciding to improve your software development processes according to CMMI® is an important step, but defining the precise improvement actions requires some structure. This document presents the method proposed by the Public Research Centre Henri Tudor that combines CMMI® and ISO/IEC 15504 to help organizations improve their processes. This paper first introduces the background of this experimentation, with ING IT Department's concern. Then it describes the method we developed and reports how we applied it in a 2-step approach in order to define a consolidated improvement plan. Finally, this paper presents discussions on lessons learnt and methodology improvement perspectives.

2 ING Luxembourg Context

ING is a global financial institution of Dutch origin offering banking, insurance and asset management services. With a network of 15 branches distributed throughout Luxembourg and the variety of its activities (retail and private banking, fund management, trading room…), ING Luxembourg needs to rely on a strong information system and on an efficient IT Department. ING Luxembourg S.A. IT Department, further referred to as ING in this paper, counts 110 persons and is organized into several entities: Application & Project Management (APM), Information Technology Services (ITS) and IT Management Support (ITMS). Mid-2007, ING and particularly the IT Management had the feeling that working habits within the IT team could be improved. They were of the opinion that relying on a worldwide standard would be an asset. They wanted to benefit from best practices collected from worldwide IT experts and to profit from standard answers provided for common problems.

In the domain of software development, several best practices standards exist (ISO/IEC 12207, CMMI® – Capability Maturity Model Integration) but ING Luxembourg rapidly opted for CMMI®, which had the advantages of being:

1) A worldwide recognized framework for implementing software development processes,

2) easily accessible, with on-line documentation and many books available,

3) already the reference at ING Group in Belgium and the Netherlands (for benchmarking).

After a period of self-training on CMMI® process descriptions and without further knowledge of CMMI® appraisal techniques, ITMS gave an attempt at assessing the IT Department's practices against CMMI® recommendations in the domain of Project Management. The lack of assessment technique was quickly a limit. So ING decided to get some directions from Public Research Centre Henri Tudor (hereafter CRP Henri Tudor), which contributes to the improvement and strengthening of the innovation capacity of organizations since 1987. It offers services for the development and experimentation of methods in information and communication technologies.

The objective was clear: ING wanted to use CMMI® as a reference for process improvement within the IT Department. The constraints were the following:

- The IT Department did not want a full CMMI® appraisal with certified results.
- They had limited resources in time and money and did not want a long and costly CMMI® implementation project.
- They wanted a pragmatic and clear action plan.
- They were open to innovative methods.

Together we established that they needed a check-up to understand where they stood and a precise work plan defining what they needed to improve. Since the CRP Henri Tudor mastered ISO/IEC assessment techniques and was eager to apply them to various domains, we proposed to combine CMMI® process descriptions to ISO/IEC 15504 assessment framework to achieve that goal.

3 Methodology

CMMI ® for development and ISO/IEC 15504

The purpose of CMMI® for development (hereafter CMMI®) is to help organizations improve their software development and maintenance processes [1][2]. It is a process improvement maturity model based on base practices. It integrates a body of knowledge on software development collected for many years by the Software Engineering Institute at Carnegie Mellon. It provides a collection of processes organized in categories (Process Management, Project Management, Engineering and Support). Each process is described in a structured manner with specific objectives and activities.

The ISO/IEC 15504 standard defines the framework and requirements for process assessment and process improvement relying on process capability [3]. Initially known as SPICE (Software Process Improvement and Capability dEtermination), it was at first limited to software development processes as described in ISO/IEC 12207. It is now applicable to any process in any domain. The process to be assessed just needs to be described in a structured manner in a Process Assessment Model (PAM) that includes a purpose, outcomes, typical activities, inputs and outputs. The way the process is executed in the organization is compared to this standard process description. Generic criteria, applicable to any process are also studied to determine process “maturity” using the 5-level capability scale of the standard.

Combining ISO/IEC 15504 and CMMI®

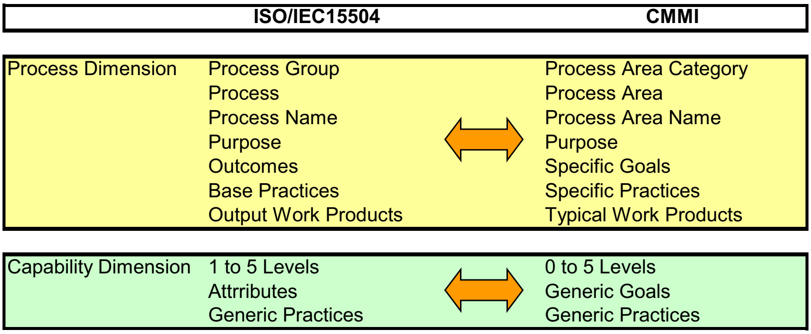
The principle of the methodology we propose is to rely on CMMI® for process definitions and to rely on ISO/IEC15504 for process assessment and improvement framework. The objectives were to:

- Keep CMMI® as the reference in the domain.
- Extract the main CMMI® recommendations and include them in a short process model thus reducing the reference documentation from a 600-page book to a 60-page Process Assessment Model - PAM
- Organize a cheap and light process assessment based on CRP Henri Tudor's previous experience [4] in the banking sector [5].
- Conduct the improvement of processes moving up levels [6] on the capability scale.

The conformance of CMMI® appraisal requirements to ISO/IEC 15504 requirements for process assessment and vice versa is clearly stated in [7]. So the assessment aspect of the methodology was easily covered. The methodology includes a standard ISO/IEC 15504 assessment [8] based on a compliant PAM.

For several years now, CRP Henri Tudor has specialized in the development of PAMs meeting ISO/IEC 15504 requirements for process description, in various domains: IT security [9], IT service management [10] [11], knowledge management [12], risk management and control in financial institutions [13]. Based on these competencies, a PAM was developed in compliance with ISO/IEC 15504 [14] to describe the processes included in CMMI®.

Over the years, both standards have evolved in parallel. CMMI® has remained specific to the world of software processes but it relies on generic principles about process assessment and capability measurement; the same principles that are specified in ISO/IEC 15504. Relying on the previous studies comparing CMMI® and ISO/IEC 15504 [15][16], a clear mapping between the concepts of both standards has been established: process areas becoming processes, specific goals relating to process outcomes, specific practices becoming base practices (See Figure 1). This mapping was the basis for the definition of the PAM.



	ISO/IEC15504		CMMI
Process Dimension	Process Group		Process Area Category
	Process		Process Area
	Process Name	⇔	Process Area Name
	Purpose		Purpose
	Outcomes		Specific Goals
	Base Practices		Specific Practices
	Output Work Products		Typical Work Products
Capability Dimension	1 to 5 Levels		0 to 5 Levels
	Attrributes	⇔	Generic Goals
	Generic Practices		Generic Practices

Figure 1 Mapping CMMI Components to ISO/IEC15504 PAM Components

Our approach

We decided to help ING following a 2-step approach. As a first step, we assessed a subset of CMMI processes using the PAM previously developed in a limited-scale project. This allowed us to determine a first list of precise actions to implement in order to improve the capability levels of these processes and to get ING's employees adhesion to this approach. As a second step, we focused on the core processes for the IT Department: the Software development domain. Their assessment leads to another list of improvement actions that were organized in a consolidated improvement plan.

Our objective was to demonstrate that the combining of ISO/IEC15504 process assessment to CMMI® process descriptions could be a solution to implement CMMI® in a rather small organization with limited resources.

4 Project Overview

The Proof-of-Concept: first assessment on Project Management processes

At ING, the overall objective was to globally improve the practices in the IT Department using CMMI as reference. To demonstrate that performing an assessment is a good starting point for an improvement program and to demonstrate the efficiency of the combination of CMMI and ISO/IEC 15504, we conducted a proof of concept. We decided to start with the project management domain and its 6 processes (Figure 2). As demonstration, we wanted to start with a limited scale assessment. A member of ITMS is dedicated to the project office function and has a good overview of all the Project Management activities and the related procedures, tools and materials existing at ING. So we decided to first assess the Project Management domain by interviewing this single person. This first assessment was conducted in December 2007. For each of the project management processes, we collected data, reviewed the documents used to perform the process and then, we analyzed and rated its underlying activities. We had determined with the assessment sponsor that the target level for the assessment was level 3 of ISO/IEC 15504 scale. Indeed, their objective was to have standard practices among the IT Department (level 3). They did not want to setup measures and indicators (level 4) or implement continuous improvement (level 5). With the interview results, we determined the level reached for each process. We also summarized Strengths, Weaknesses, Opportunities and Risks (SWOR) related to each process in an assessment report, and suggested a first improvement plan, which contained 41 recommendations.

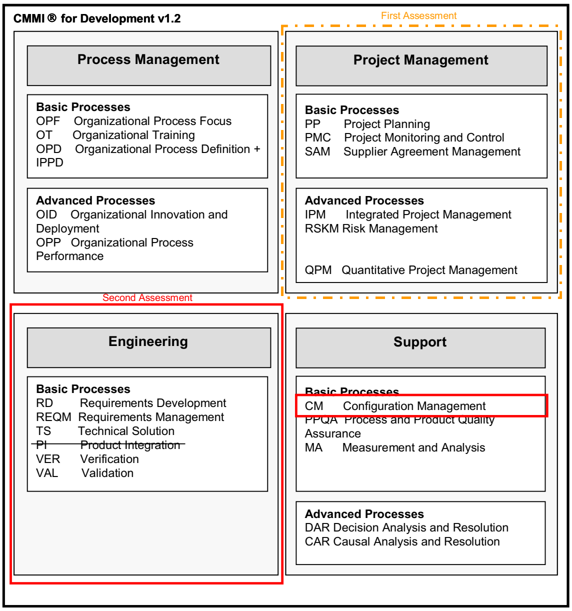

CMMI ® for Development v1.2
First Assessment
Process Management
Basic Processes
OPF Organizational Process Focus
OT Organizational Training
OPD Organizational Process Definition + IPPD
Advanced Processes
OID Organizational Innovation and Deployment
OPP Organizational Process Performance
Project Management
Basic Processes
PP Project Planning
PMC Project Monitoring and Control
SAM Supplier Agreement Management
Advanced Processes
IPM Integrated Project Management
RSKM Risk Management
QPM Quantitative Project Management
Second Assessment
Engineering
Basic Processes
RD Requirements Development
REQM Requirements Management
TS Technical Solution
~~PI Product Integration~~
VER Verification
VAL Validation
Support
Basic Processes
CM Configuration Management
PPQA Process and Product Quality Assurance
MA Measurement and Analysis
Advanced Processes
DAR Decision Analysis and Resolution
CAR Causal Analysis and Resolution

Figure 2 Selection of processes on the CMMI process map

The recommendations were reviewed by ING internally and were prioritized according to several criteria (relevance, urgency, expected added value, ease of implementation, impacted teams). Early 2008, ITMS started to implement some recommendations. To receive everyone's support, the more practical ones (templates, documents diffusion, practical guides, communication actions) have been followed. The implementation of "quick wins" was important to carry on with the improvement spirit that arose with the presentation of the assessment results.

The result of this first assessment was the proof of the viability of our methodology: the rating of CMMI processes, using ISO/IEC 15504 framework, was established and we provided ING with a precise list of improvement actions to better implement CMMI. 10 days were spent by CRP Henri Tudor and around 6 days were required from ING. A 53-pages assessment report and a 65-slides results presentation were produced. The methodology proved cost-effective and provided the expected results, which ING considered of high professional quality.

Second assessment on Engineering processes

For ING, the proof of concept was conclusive and, in September 2008, the second step of the global assessment started. Its purpose was to assess and rate the processes of the Engineering domain in order to establish the basis of a future global improvement program for the IT Department. During the initial phase of assessment scope definition, we established that the Product Integration process was not applicable in the context of ING. We also decided to cover the Configuration Management process (from the support domain) since it is closely related to development activities (Figure 2).

Helped by ITMS, we chose, for each process, the key related people and planned interviews with them. End of September, a kick-off meeting was organized, with involved persons (sponsor, interviewees, and assessment team members) in order to explain the objectives of the approach, the methodology used, and the practical aspects of the assessment.

Over 3 weeks, 3 assessors conducted 12 interviews on the 6 processes for a total of 25 hours with 12 persons. The plan was to interview persons from the APM team, but after the first interviews on the Validation process, we realized we needed to see other persons, from ING business departments, and closer to day-to-day tasks on this process, in order to complete our overview. In a general manner, we gathered all kinds of useful information provided by employees during these 'open-mode' interviews.

The analysis of the assessment results lasted another 3 weeks. We first agreed on the rating and conclusions of each individual interview. Then for each assessed process, we determined its capability level according to ISO/IEC 15504 standard, we highlighted its strengths, weaknesses, opportunities and risks and we proposed 57 recommendations to cover gaps compared to the assessment target and improve the process maturity up to level 3. These results and the improvement plan drafted from the recommendations were summarized in the assessment report and presented to all involved persons.

A consolidated improvement plan

The first assessment at the end of 2007 resulted in a first improvement plan for Project Management processes validated by the IT Management. It was followed by several improvements conducted by ITMS. After the second assessment, with the new list of recommendations and improvement plan for engineering processes, ING needed some guidance on the organization of the overall improvement program for IT. We helped them to fix priorities using the same criteria as previously and to define a consolidated improvement plan, taking into account both assessments findings and recommendations as well as the improvements implemented in the mean time. The whole 2-step approach is summarized in Figure 3

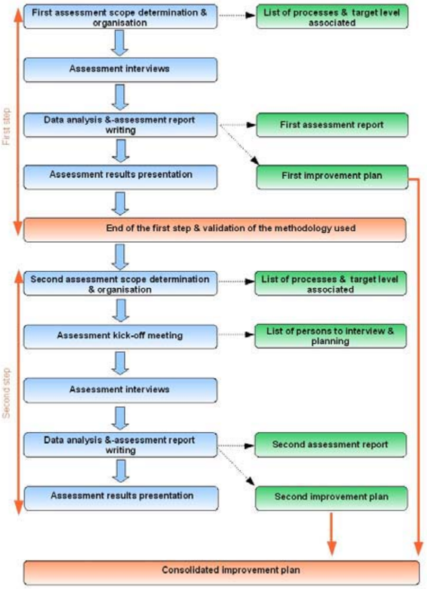

First assessment scope determination & organisation
List of processes & target level associated
Assessment interviews
First step
Data analysis &-assessment report writing
First assessment report
Assessment results presentation
First improvement plan
End of the first step & validation of the methodology used
Second assessment scope determination & organisation
List of processes & target level associated
Assessment kick-off meeting
List of persons to interview & planning
Second step
Assessment interviews
Data analysis &-assessment report writing
Second assessment report
Assessment results presentation
Second improvement plan
Consolidated improvement plan

Figure 3 Improvement approach Diagram

The proposed program spans over a period of 24 months in order to achieve capability level 3 for processes of Project Management and Engineering domains. The implementation timing will of course depend on the priorities fixed by ING as well as on the resources dedicated to this program and on the involvement of stakeholders.

5 Lessons learned

First conclusions on the approach

Practices at ING have been compared to a reliable reference framework, without the drawbacks of a long and expensive approach. As the certification was not an end in itself, we have proposed a light way, well adapted to ING's requirements (reduced time and resources, CMMI as reference, effective action plan) to simplify and quickly initiate a structured improvement program through an efficient process assessment.

The combination of CMMI® and ISO/IEC 15504 has proven a good framework to assess and improve process capability. Indeed, the methodical nature of ISO/IEC 15504 provides a way to structure the improvement and improve practices with the help of recommendations from CMMI®. This approach does not require a certified CMMI® assessor, which undeniably simplifies, lightens, accelerates and reduces the cost of the approach.

In the domain of software process improvement, this experience meets up with other points of view [17] [18] concluding that CMMI® is a strong reference that is also applicable to SMEs. CMMI® process descriptions and recommendations are generic and high-level enough to serve as good practice in an organization that does not have major resources available. We have also noticed that despite the specificities of IT in the banking sector (legacy systems, integration of packages), the approach was still generic. ISO/IEC 15504 assessment and improvement framework as well as CMMI processes are applicable in that context, which helps us conclude that this combination could be applied to other domains.

Key factors to succeed with this approach

By reviewing the ING experience, we have highlighted some key success factors. These factors, even if typical to improvement projects, are nevertheless critical and deserve to be stressed as follows:

1 – To have a strong sponsorship from the management. The stronger the sponsorship is, the higher the awareness of the staff is. The higher the awareness is, the more important the involvement is. And if employees feel involved, they have a mission and they will adopt a constructive attitude and will be more inclined to suggestions. In fact, sponsorship is the best way to limit resistance to changes.

2 – To manage the improvement project. It is necessary to have a real improvement project with objectives, deadlines, budget and regular status report to the management. A project manager should be appointed too. A person must be officially mandated to conduct the work plan. This is essential for the success of the project because with this mandate, this person will have the authority to ask everyone to contribute and to regularly assess and follow up the project progress.

3 – To bring out strengths. The assessment conclusion should not merely reveal the gaps in the practices. It should also underline the positive aspects. One goal is to make an inventory and standardize good practices and to feed the improvement plan with existing practices.

4 – To identify quick wins. For the good spirit of the team, you should define some results, which will be visible and easy to reach. The actions in the improvement plan must be as short as possible. Thus, it will be possible to regularly announce new results. These announcements will also encourage the team and keep motivation alive.

5 – To adopt a progressive and realistic approach. Even if many processes do not seem to work well, one should not try to improve all processes at the same time. Indeed, such an initiative could be perceived as a big bang and could be frightful for the teams. It is best to progress gradually and iteratively. The work plan must also be in line with the ressources that can be dedicated to it. An unrealistic plan could discourage and demotivate people, especially when they will become aware of the extent of the work.

6 – To communicate. An improvement approach really needs a lot of communication actions: presentations, progress meeting, working group, training or coaching. The objective is to keep the connection with people that are implied and to inform them of the progress of the project.

Possible improvements for our approach

We have also identified some aspects that could be improved in our approach:

To reinforce the preparation phase. We prepared the process assessment with the IT Management to understand the context at a high level. But we did not review in details what existed to support the software engineering process in the IT department. With such a review, the assessment could have been more efficient, particularly for the third capability level. Indeed, some interviewees were not aware of the existence of standard tools, templates or procedures. We concluded that nothing existed at the corporate level, but after several interviews, we heard of the existence of some standard elements. Processes were highly standardized with templates and tools but the deployment of these standard processes was weak. The lack of detail in our initial review could have distorted the assessment results and have led us to wrong improvement recommendations because we would not have detected that standard tools, templates and procedures did exist but were not properly deployed.

So the proposed change to improve our approach is to first make an inventory of existing tools and procedures, particularly if there is a reference framework in the organization.

To pay attention to the selection of interviewees. The second point that could be improved is the way the organization selected the panel of interviewees. One of the selection criteria we gave to ING was to choose operational people. But a natural tendency leads to choose managers because they are supposed to have a good overview of what is realized in their team. But the experience proves that they indeed know what is realized, but they often have no idea about how it is realized. So they give answers about the way they assume things are realized, and the good or bad practices that could exist in their teams are not taken into account. At ING, we did not hesitate to reschedule some interviews with additional people, by insisting on the fact that they had to be operational people.

In conclusion, we should insist on the selection criteria and speak up if we think that there is a problem with the panel composition.

To include Business. During this project, we encountered a problem concerning the perimeter of the assessment. Our objective was to assess the maturity of software engineering processes. So we performed interviews within the IT department. But as the mission of IT is to support the business, IT department must work closely with business teams. In this context, interviewing only IT people cannot conduct to a complete picture. On the one hand, important practices of processes cannot be assessed (for example practices concerning the way the business insures the validation of the application). On the other hand, improvement actions (new templates, new procedures), which impact the business, will be hard to implement and to get accepted by people that were not involved in the assessment approach.

This leads us to systematically propose to involve in the assessment some of the business teams involved in the project as “customers” of IT department.

6 Conclusion

The combination of CMMI® and ISO/IEC15504 we proposed allowed ING to achieve their objective: compare themselves to a worldwide standard and define their improvement program according to CMMI®. The project achieved its objectives within the limited time and resources.

Considering the results that we obtained, this approach seems to be an efficient way to make CMMI® accessible for organizations that do not have the means to launch a large CMMI® implementation program. Of course, the objective is not to help organizations obtain CMMI® certification, but to help them improve their practices by animating a collaborative improvement approach.

The ING IT Department’s satisfaction encourages us to continue working on this approach. Particularly we would like to work with other organizations to further experiment and develop this method. The main improvement opportunity is to strengthen the preparation and specially to convince IT managers to include the business departments in its improvement initiative. Indeed, it is not realistic to believe that IT can improve its processes without considering also business practices related to IT projects. If we succeed in systematically convincing managers to include both IT and business in assessment and improvement projects, it will help to have a better picture of the real strengths and weaknesses of the organization. But it will also help gloss over the boundaries between IT and business, which is one of the major obstacles for efficient IT governance.

To conclude, we would like to emphasize that our method is particularly well suited for the Luxembourg context, where most organizations are small or medium-sized but belong to a worldwide group. In the Banking sector, IT Teams are small but they are expected to follow the major quality standards that are defined at the group level (CMMI®, ITIL). Our CMMI® conformant process assessment model constitutes a good reference material with an acceptable scale for these reduced teams. Our method provides them with a pragmatic approach to process improvement and a way to align to their group's requirements regarding compliance to major standards. We plan on further experimenting our approach in a similar context this year.

7 Literature

1. CMMI product team "CMMI® for development, Version 1.2: Improving processes for better products", Carnegie Mellon, Software Engineering Institute, August 2006.
2. Mary Beth Chrissis, Mike Konrad, Sandy Shrum "CMMI® Guidelines for Process imtegration and Product Improvement", Second Edition, Addison-Wesley, 2007.
3. Terence P. Rout, Khaled El Emam, Mario Fusani, Dennis Goldenson, Ho-Won Jung: SPICE in retrospect: Developing a standard for process assessment. Journal of Systems and Software 80(9): 1483-1493 (2007)
4. Béatrix Barafort, Jean-Philippe Bodelet "SPI Case Study in a Direct Response Marketing Company", EuroSPI'2003, Graz, Austria Dec. 2003
5. Hauman Steve, "Using ISO/IEC 15504 to improve the IT development process in a banking context", Keynote in SPICE 2006 conference, Luxembourg, 2006.
6. Jean-Philippe Bodelet, Mathieu Lambert, Samuel Renault, "Managing Change Generated by a SPI Project in a Banking Organization", SPICE'2008, Nuremberg, Germany, May 2008.
7. SCAMPI upgrade team "Appraisal Requirements for CMMI®, version 1.2 (ARC, V1.2), Carnegie Mellon, Software Engineering Institute, August 2006.
8. ISO, ISO/IEC 15504-2: Information technology - Process assessment - Part 2: Performing an assessment, 2003
9. Barafort B., Humbert J-P., Poggi S., Information Security Management and ISO/IEC 15504: the link opportunity between Security and Quality". SPICE '2006 conference, Luxembourg, May 2006.
10. Barafort B., Di Renzo B., Lejeune V., Simon J-M, ITIL Based Service Management measurement and ISO/IEC 15504 process assessment: a win – win opportunity, SPICE'2005, Klagenfurt, Austria, 2005
11. Hilbert R., Renault A., Assessing IT Service Management Processes with AIDA – Experience Feedback, EuroSPI'2007, Potsdam, Germany, 2007
12. Di Renzo B., Valoggia Ph., "Assessment and Improvement of Firm's Knowledge Management Capabilities by using a KM Process Assessment compliant to ISO/IEC 15504. A Case Study", SPICE'2007, Seoul, South Korea, May 2007.
13. Di Renzo B., Hillairet M., Picard M., Rifaut A., Bernard C., Hagen D., Maar P., Reinard D., "Operational Risk Management in Financial Institutions: Process Assessment in Concordance with Basel II", SPICE'2005, Klagenfurt, Austria, April 2005
14. ISO, ISO/IEC 15504-5: Information technology - Software Process Assessment - Part 5: An exemplar process assessment model, 2006
15. Terence P. Rout, Angela Tuffley, "The ISO 15504 Measurement Framework for Process Capability and CMMI", SPICE'2005 Klagenfurt, Austria, 2005
16. Terence P. Rout, Angela Tuffley, "SPICE and CMMI: Conformance of the CMMI Models to ISO/IEC 15504", SPICE 2002, Venice, Italy, March 2002.

17. F. G. Wilkie, D. McFall, F. McCaffery “An evaluation of CMMI process areas for small- to medium-sized software development organisations“ Software Process: Improvement and Practice, 2005 - interscience.wiley.com

18. SZ Garcia “Thoughts on Applying CMMI in Small Settings” Retrieved October 9th, 2008 - sei.cmu.edu

8 Author CVs

Valérie Betry

Project manager at CRP Henri Tudor (Luxembourg) since 2003. Graduated from Nancy Business School and Master in Information System Design from the University of Nancy2. Worked as developer then IT business analyst within Arthur Andersen internal IT team for 5 years.
At CRP Henri Tudor, involved as project manager on research projects related to process assessment and improvement as well as on software development projects. Consulting assignments for private clients on process assessment, process improvement, project management method implementation and software package selection. Major competences include IT project management, quality assurance and software process assessment and improvement.

Stéphane Cortina

R&D Engineer at CRP Henri Tudor (Luxembourg) since 2007. Graduated from the University of Metz and Master in Software Engineering from the University Joseph Fourier of Grenoble, Stéphane is also ISO/IEC 15504 Assessor, member of Luxembourg chapter of itSMF and of ILNAS (The Luxembourg Institute for Normalization, Accreditation, Product Safety and Quality). At CRP Henri Tudor, he is involved on research projects related to normalization, IT service management and process assessment and improvement.

Séverine Mignon

Unit manager at CRP Henri Tudor (Luxembourg) since 2000. Master in Multimedia Engineering from the University of Metz. At CRP Henri Tudor, worked first as organization and communication manager. Then involved as project manager on research projects related to software development. Actually manager of a software engineering competences unit, specifically in charge of software quality aspects, from process quality level to product quality level. Major competences include IT project management, quality audit and software quality assessment and improvement.

Two Years Defining and Implementing Software Processes: The Experience of a Brazilian University

Adriano Bessa Albuquerque[1], Nabor Chagas Mendonça[1], Carla Ilane Moreira Bezerra[1], Shamya Ribeiro Gomes[1], Alexandre Lima Gondim[1]
[1] University of Fortaleza (UNIFOR), Washington Soares Avenue, 1321 - Bl J Sl 30
ZIP 60811-341 – Fortaleza, Brazil
{adriano.ba@terra.com.br, nabor@unifor.br, carlailane@yahoo.com.br,
shamyarg@gmail.com, alexandrelima21@gmail.com}

Abstract. The importance and opportunities of the software industry is motivating software organizations to improve their products and consequently, to define and implement software processes able to achieve this objective. However, these activities are not trivial and must be conducted considering many factors. Therefore, specialized consultancies that assist software organizations with this challenge should reflect about their experiences and formalize an implementation process to support all necessary actions. Moreover, any implementation process must be adaptable and constantly improved. This paper presents some lessons learned captured during two years of experience defining and implementing software processes at a Brazilian university, consultancy of MPS.BR. In addition we define an implementation strategy after some reflections concerning these experiences.

Keywords: Software Process Improvement, Software Definition, Software Deployment.

1. Introduction

Nowadays, the software industry is one of the most promising of the world [1]. Software is present on almost all products and services and sometimes the existence of a company and its growth depend a lot on the quality of their software products.

However, developing good software can not be seen as magic, that is, the organization must carry out some initiatives to obtain such result. One of the most relevant is the implementation of software processes. Defining and implementing software processes are not trivial and must be treated adequately, because many factors influence the success of these initiatives, such as those related to human resources, financial resources, and knowledge. Besides, each organization should have different strategies given its organizational culture, characteristics and conditions.

This paper presents some lessons learned during two years of consultancy, defining and implementing software processes. We briefly describe a new implementation strategy to define and implement software processes for our customers.

Section 2 presents some success factors of Improvement Programs. Section 3 presents UNIFOR's software processes which are the basis do adapt clients' processes, Section 4 presents some lessons learned and an overview of the actual UNIFOR's implementation strategy. Section 5 concludes this work.

2. Improvement Program

The formalization of an Improvement Program is fundamental to support software organizations implementing software processes. We can find many aspects that influence positively the results expected by an Improvement Program.

Furthermore, others aspects must be considered adequately [2], [3], [4], [5], [6]. For example, the organization must provide sufficient resources (financial, human, and technological) to supply what is necessary and to guarantee the continuity of the Program. Another important initiative is to frequently analyze the return on investment with the improvements that were implemented and the financial resources that were already invested on the Improvement Program. This action can motivate high management to continue investing on the Improvement Program Project. Additionally, it is also very important to see the program as a long-term investment. This is essential for its continuity, to avoid high management of having false expectations concerning the speed of return.

The Improvement Program must be customized according to some characteristics of the organization, specially its culture, type of projects and business. When the Improvement Program is adapted, it is more likely that the processes will provide support in a better way to the organization's business. Thus the improvement objectives must be adjusted to its business strategies objectives.

Another important conduct is not to deal only with technical factors on the definition and actions of the Improvement Program. Human resources and organizational culture must be considered in all relevant points of the Improvement Program. For example, investment on qualification of human resources can be seen as one of the most important success factors. Finally, collaborators engagement is decisive as well as support provided by knowledge management approaches.

3. Experience of a Brazilian Consultancy in Defining and Implementing Software Processes for Software Organizations

UNIFOR is a large university from Brazil and is one of the official consultancies of the MPS.BR, the Brazilian Software Maturity Model [7]. In 2007, UNIFOR began to define the following processes: Project Management, Requirement Management, Configuration Management, Quality Management and Measurement and Analysis for three software organizations of Fortaleza, a city in Northeastern Brazil.

To define the processes, firstly a meeting was held between UNIFOR and each organization to obtain commitment of all stakeholders. During this meeting were introduced the responsibilities of all participants and a project plan for the Improvement Program. At this point commitment with the work plan was obtained.

On another day the consultants of UNIFOR went to the organizations to evaluate their actual processes, performing a gap analysis. The processes were evaluated according to the expected results and processes attributes defined by the reference model of the maturity model (MPS.BR).

During the evaluation, documents describing the actual processes and guidelines for their software development were analyzed. Afterwards, some interviews were held with analysts, programmers, project managers and others members of the technical team. When the evaluation finished, a document with the processes diagnosis was presented and handed over to the organization. After the diagnosis, UNIFOR's consultants began to define the processes, considering non-compliances found during gap analysis: characteristics of the organizational culture and of the technical, personal and organizational structure. Then the processes were introduced to the organizations and some modifications were implemented.

The implementation of the processes comprised training the organization's collaborators in the new processes and selecting one or more pilot projects. The collaborators had trainings in each process that would be implemented and the duration of each course was between four and eight hours. The pilot project was selected by a consensus between the organization's high managers and the consultants.

When the selected projects began, each consultant visited the organizations two times a week. The consultants constantly collected data to monitor the progress of the implementation. Plus at the beginning of the implementation, once a week a meeting was held with all consultants, aiming to monitor the implementation, disseminate relevant knowledge, and identify weaknesses and strengths of the consultancy and barriers in the organization.

At that moment were also held meetings with the consultancy and collaborators to help them to understand some important points: (i) the relevance of initiatives of process improvement for the organization, (ii) the importance to execute the processes as defined and (iii) the importance for each collaborator's career to participate actively in an Improvement Program, because it is an opportunity to learn Software Engineering and to become more employable.

Some meetings were held with the process groups and top management aiming to define new strategies of implementation, identify implementation barriers and define improvement opportunities for the implementation process. During implementation the processes continued being improved: new tools began to support the execution of some activities, templates were modified to become more adequate to the organizations' needs and some people were reassigned to new roles.

All organizations were assessed on the MPS.BR in 2008. One of them obtained level G (Project Management and Requirement Management) and F (Project Management, Requirement Management, Quality Management, Configuration Management and Measurement and Analysis). Another one obtained level G and the third obtained good results on the initial assessment in level F in December 2008.

4. Lessons Learned in Two Years of Definition and Implementation of Software Processes

At the beginning of 2009, the consultancy leaders – two professors of UNIFOR – began to think about the rich experience in defining and implementing software processes in Brazilian software organizations. These experiences involved companies with different cultures, objectives, knowledge, skills of professionals, perceptions of quality values and so on. Additionally, concerning the consultancy's characteristics, we had different consultants, with different levels of knowledge, involvement, experience and ability to assist clients.

This initiative resulted in a list of lessons learned related to the definition and implementation of software processes which will help us for next efforts and specially will guide other consultancies. On the 2nd Semester of 2007, when we began to implement the processes we defined a strategy where the consultants presented the processes and the templates to the collaborators, demanding them to fill out the templates to be evaluated in a next session.

Lesson Learned 1 – this was not an effective strategy because when we came back, the activities had not been executed. Therefore, we needed to change the way we were performing the consultancy. We began to help more intensively executing the activities, even filling out the templates with them.

Moreover, at the beginning, we decided that the consultants would assist all organizations and they would be involved in all projects.

Lesson Learned 2 - this strategy had a lot of problems. One of them was the difficulty to supervise the activities of the consultants and to know what exactly they were doing. Another deficiency was the effort necessary to maintain an adequate level of communication between consultants. Because of these problems, we improved the way we were working. Each consultant became responsible for specific projects and the responsibilities were not shared anymore. This change increased the effectiveness of the consultancy.

During the 1st Semester of 2008, one of the organizations had its first assessment on MPS.BR, named "Initial Assessment", which is similar to the Readiness Assessment from CMMI.

Lesson Learned 3 – we observed that this initial assessment was very important for the organization and for the consultancy, because all those involved learned a lot with the experience and the different and external points of view of the auditors. Moreover, the success of this assessment became a great motivator also to the other clients.

The project managers had to constantly learn Software Engineering and work a lot, because many of the activities depended, in some way, of their initiatives.

Lesson Learned 4 – The engagement of project managers is fundamental and is one of the most important success factors of an Improvement Program. We always analyzed if they really could be managing the project. In one of the organizations, the project manager was changed during the execution of the project. This action improved considerably the level of adherence to the processes.

A lot of meetings were held with the collaborators responsible for implementing the Improvement Program to identify what could be improved and what were the main barriers to institutionalize the processes in the organizations.

Lesson Learned 5 – the posture of the Improvement Program leaders is another very important success factor. They must have an attitude of leadership and have a good relationship with the various stakeholders (high management, project manager, development team, consultants etc).

Some organizations did not have the level of engagement we expected and they worked very slowly. However, after one of these organizations scheduled the initial assessment we noticed that the level of involvement of the collaborators increased considerably.

Lesson Learned 6 – If an organization has good chances to be successfully assessed, scheduling an assessment is an interesting strategy to obtain the adhesion of collaborators. Besides, the consultants also become more engaged with the Improvement Program.

Finally at the 2nd Semester of 2008, when the organizations began to be assessed, implementation at other clients became faster and risks of failure decreased a lot.

Lesson Learned 7 – It is very important for the consultancy to organize their assets, aiming to reuse them in others organizations and situations to accelerate the implementation of the processes. We could also observe that when the gap between the definition of the processes and their implementation is very large, the motivation of the organization tends to reduce.

In August of 2008 one of the organizations obtained the F level, implementing Project Management, Requirement Management, Quality Management, Measurement and Analysis and Configuration Management. However, the consultancy had a lot of difficulties at assisting the organization to obtain this level.

Lesson Learned 8 – The necessary effort of the consultancy needed to institutionalize the processes from level F is much bigger than those from level G, because, mainly, new roles are demanded and each new process must interface with the old ones. Likewise, the organization must provide an ideal structure for this implementation.

When one of the organizations decided to implement the processes from level F, they had to contemplate new tools to provide support to the processes execution.

Lesson Learned 9 – In the advanced levels, organizations must invest in the acquisition of new tools to enhance the productivity of executing the process and to allow a better interaction between diverse processes of an organization. From our experience, especially the Configuration Management process must be sustained with adequate software solutions.

Lesson Learned 10 – The Team Foundation tool can be considered a good solution to support the processes when the organization is a Microsoft partner, because it becomes cheaper and it can support the management of demands and of activities related to Configuration Management.

In one of the organizations, the organization had few collaborators and therefore we had to define many roles to only one collaborator.

Lesson Learned 11 – when only one collaborator performs many roles, probably, the quality of executing their activities will be influenced negatively, because he will not have time to reflect deeply on fundamental issues and decisions.

In two organizations the allocated project managers had not sufficient experience and had recently finished their undergraduate course. Sometimes, that was a big challenge for the consultancy.

Lesson Learned 12 – Comparing the organizations where the project managers have experience with those with inexperienced managers, we could see that the level of experience has a positive influence on the quality of work products elaborated and on the decrease of the consultancy’s effort.

Lesson Learned 13 – The Quality Management process must be defined and implemented carefully, because if this process is not well implemented, it will jeopardize the quality of other processes and of the final software product.

In another organization, when we began to define the processes we found a new and different scenario. All the participants of the process group were administrators and did not have experience in software engineering. We had a lot of difficulties to define the processes because they did not understand software engineering concepts and did not visualize the activities involved with software development.

Lesson Learned 14 – the roles involved in the Improvement Program must be performed by professionals with experience in software engineering. This makes the communication easier with the consultancy and the processes can be more adapted to the organization’s characteristics. Besides, non-experienced professionals will not succeed in executing the activities adequately.

An organization requested the improvement of its agile process (SCRUM) so that it could become adherent to level F of MPS.BR. During the first meeting we were able to understand that the request would not be a challenge and that we would easily specify the necessary improvements.

Lesson Learned 15 – When organizations that already use agile processes want to be assessed on some maturity model it is very important to spend time, initially, on meetings to understand their culture and processes and to make clear the changes that will be implemented, because this kind of organization has discrimination with control approaches and the existence of artifacts.

Lesson Learned 16 – When the organizations were certified in ISO 9000 the definition and implementation of processes became easier because they had already the culture of processes and artifacts.

After the identification of the lessons learned described above, we classified the organizations using the following criteria, captured from [2], [3], [4], [5] and [6]: (i) financial support; (ii) quality of human resource; (iii) long term vision of the Improvement Program; (iv) collaborators' engagement; (iv) knowledge management support; (v) investment in improving staff's competences and skills; (vi) horizontal organizational structure; (vii) motivation and cooperation of employees; (viii) policy of recognition for those who collaborate with the Improvement Program; (ix) respect the consultancy's advices; (x) understanding of the Improvement Program's benefits; (xi) high level of turnover; (xii) support of top management; and (xiii) flexibility to incorporate changes.

When we analyzed the reality of our clients we could observe that our experience confirmed some of the success factor listed above. The top management of the organization that was assessed successfully in two levels, during only one year, had a long term view related to the Improvement Program, invested financial resources and hired competent collaborators to perform important activities. However, some collaborators could not participate of the second assessment and it was a serious problem. One of the collaborators told us that his low level of motivation was related to the fact that the organization did not provide any additional benefit to him, who had been engaged a lot on the first assessment. Moreover, because of the understanding of the collaborators about the relevance of the Improvement Program, all the consultants were very well received and their advices were almost always considered and implemented.

On a second client, one of the barriers to implement the processes was the fact that they knew the relevance to improve the processes but it was not very clear the importance to be assessed on a maturity model. So, they worked as they had all the time of the world to define and to execute the processes during the projects. Its organizational structure, in spite of being horizontal, supporting the communication between the collaborators, influenced negatively the Improvement Program because some decisions were not performed quickly neither deeply, since the members of the top management were performing technical activities.

The third organization had a lot of problems to implement the new processes due to the low level of qualification of the collaborators involved on the projects. This company also chose not to invest on the improvement of their staff, and not to hire experienced professionals to perform the relevant activities. The motivation for this decision could have been the reduction of the projects' cost. In relation to the organizational structure, the main problem, as they develop embedded systems, was the difficult of communication between the development team and the hardware team. Finally, we could observe that when an important business opportunity came, some resources were reallocated to this opportunity.

After these experiences and thinking about the weaknesses of the way we were defining and implementing processes for our clients, we decided to define a new strategy to implement processes, aiming to deal with all relevant factors which should be considered. The definition of the strategy was based on the PDCA Cycle and comprises four activities. We decided to guide this strategy by the assumption that an Improvement Program must be seen and considered as a project. Follows a brief description of each activity:

(1) Plan the implementation: The purpose of this activity is to plan the implementation strategy, considering the characteristics of the organization and define the necessary resources to institutionalize the processes. It comprises the following tasks: **(i) identify the client,** where the improvement objectives are identified; relevant organization's characteristics are identified and a gap analysis is carried out; **(ii) adapt processes,** where some activities or approaches are inserted or removed from the consultancy's standard processes; the team of consultants are defined; the main risks are defined and training needs to the consultants are identified; **(iii) elaborate the process,** where the implementation process is adapted according to the organization's characteristics; the scheduled is defined; the client's participants are defined and the milestones are defined.

(2) Execute the implementation: The purpose of this activity is to execute the defined implementation process and to collect relevant data. It comprises the following tasks: **(i) hold a motivation meeting ; (ii) present the defined process to the organization's collaborators**; **(iii) conduct formal trainings**; **(iv) conduct hands-on trainings**; **(v) redefine the processes' activities**; **(vi) redefine templates**; **(vii) conclude the consultancy,** when a post-mortem meeting is held.

(3) Verify the implementation: The purpose of this activity is to analyze the obtained results. It comprehends the following tasks: **(i) conceive meetings to monitor implementation**; **(ii) monitoring the implementation on milestones; (iii) conduct informal assessments**.

(4) Improve the implementation strategy: The purpose of this activity is to define and implement corrective actions for the implementation process. It comprises the following tasks: **(i) define corrective actions plan**; **(ii) manage the corrective actions**; **(iii) execute the corrective actions**.

5. Conclusion and Further Works

The definition and implementation of software processes are fundamental for software organizations. However these are arduous tasks for both clients and consultancies. In two years of work, as a consultancy, we could identify many barriers and lessons learned, which, if understood and well reflected, they can be incorporated into new strategies of processes' implementation. However, there are more ways to define and implement processes, because it depends on the customer's characteristics and any implementation strategy must remain in constant state of continuous improvement. As further work, we will validate the described implementation strategy in a case study and we are going to promote a Client Forum to capture improvements to be performed on our strategy.

References

1. Nollen, S.: Software Industry Performance in India and China, In: INDIA in the Emerging Global Order, eds. Amar K J Nayak and M G Jomon, Tata McGraw-Hill Publishing, Chapter 4 (2008).
3. Dyba, T.: Factors of Software Process Improvement Success in Small Organizations: An Empirical Study in the Scandinavian Context, In: Proceedings of the ESEC/FSE'03, pp. 148-157, Helsinki (2003).
4. Hefner, R., Tauser, J.: Things They Never Taught You in CMM School, In: Proceedings of the 26th Annual NASA Goddard Software Engineering Workshop, pp. 27-29, November (2001).
5. Albuquerque, A. B.: Evaluation and Improvement of Organizational Process Assets in Software Development Environments, Ph. D Thesis, COPPE/UFRJ, Rio de Janeiro, RJ, Brazil (2008).
6. Montoni, M.: An Approach to Conduct Initiatives of Software Process Improvements, Qualification Exam, COPPE/UFRJ, Rio de Janeiro, RJ, Brazil (2007).
7. Debou, C.: Goal Based Software Process Improvement Planning, In: Messnarz, R. and Tully, C. (eds.), Better Software Practice for Business Benefit: Principles and Experience, cap. 5, IEEE Computer Society Press (1999).
8. SOFTEX: Brazilian Software Process Improvement – General Guide version 1.2, http://www.softex.br/mpsbr.

Combining EXAM with model-centric testing

Dr. Anne Kramer

Abstract

In this paper we present an improvement case study for testing. We combined the EXtended Automation Method (EXAM) of the Volkswagen AG and Audi AG with principles of model-centric testing. EXAM is a strategic project at Volkswagen AG and is used throughout the entire corporation. It is designed to standardize test methods and to increase the re-usability of tests via test libraries. To this well established concept we added aspects of model-centric testing by introducing behavior models and automatic test case generation.

We explain in detail the current approach, the difficulties that have been addressed, how they could be improved and what are the next steps.

Keywords

EXAM, model-centric test, test case generation

1 Introduction

The EXtended Automation Method (EXAM) is a strategic project of the Volkswagen AG to provide a highly productive platform for joint development of test programs for electronic control units. It allows Volkswagen AG and Audi AG to establish not only a common test strategy, but also a common language for describing tests throughout the entire corporate group [1]. Its main goal is to increase the standardization and reusability of tests.

The test automation method combines the principles of modeling techniques with test automation. Test cases are mainly described as sequence diagrams which are partly assembled from existing sub-diagrams stored in libraries. The test cases are then parameterized, combined to test suites and test campaigns and finally translated into test applications that are executed on a hardware-in-the-loop test system.

In this paper we describe, how EXAM has been combined with aspects of model-centric testing. Model-centric testing is a method to describe test artifacts with models (e.g. activity or state diagrams), based on behavior models and taking the tester's mindset into account. (For a more detailed description of model-centric testing, see also [2] and [3].) Unlike in model-based testing, the model contains all information relevant for the test. This includes test management information as well as test data. The model constitutes the basis for all discussions and is an essential part of the documentation.

In 2007 sepp.med proposed to introduce the principles of model-centric testing to EXAM at Audi AG in Ingolstadt. Together with Audi, the concept has been enhanced to include behavior diagrams that describe the system under test. Using the test case generator .getmore, the sequence diagrams that correspond to the test cases are now generated automatically from the models.

2 Case study

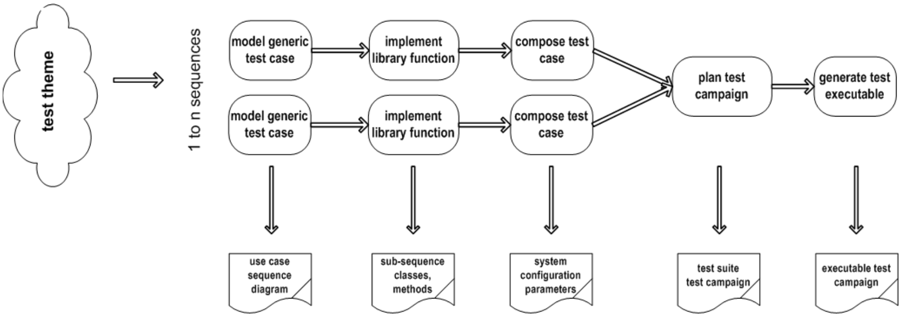


Fig. 1: Test case design and implementation with EXAM

2.1 The situation in the beginning

2.1.1 Testing with EXAM

The test workflow can be divided into two major phases:

- test design and implementation,
- test execution and analysis of results.

The first phase is shown in figure 1. With EXAM the majority of tests are systematically designed as sequence diagrams. These diagrams describe the test sequences the test author has in mind. They are partly assembled from existing sub-sequences. These sub-diagrams are stored in test libraries and managed by EXAM. For new functionality new sub-diagrams are written and stored in the libraries.

The output of the test design activity is a variety of sequence diagrams. The EXAM code generator translates the test sequences in Python code that is linked with atomic, hand-coded code bodies.

The sequence diagrams correspond to generic test specifications. The test cases are not yet parameterized for the specific product variant or test system that shall be used and tested. This is done separately when the test case is "composed".

Next, the test campaign is planned. The test campaign is a set of one or more test suites that contain all test cases that shall be executed in this particular test run. The test campaign is an automated test script that can be interpreted and executed with Python.

Besides the last step, i.e. the generation of the executable, this is done manually.

In the second phase, the test campaign is automatically executed by the hardware-in-the-loop test system. The test system logs the results for further analysis. The results are then compared to a previously defined expected result given for example by the parameterization of the test cases. This expected result must be maintained manually.

2.1.2 Improvement potential

EXAM has proven to be an efficient way to standardize tests and to increase the re-usability. Still, there are certain difficulties that persist with EXAM (even if they would be much worse without EXAM).

The writing of test cases in form of sequence diagrams is still rather time consuming and error-prone. There is no systematic check of the completeness of test cases, especially if it comes to error situations or different user behavior.

The test design is done in a rather unguided way, strongly depending on the tester’s mindset and experience. The reasoning, what has been tested and why the test case is organized the way it is, stays undocumented. Thus, it is difficult for others to understand what the test case author had in mind. Also, decisions, what has been left out in a test campaign and why are not documented.

Discussion with other stakeholders is difficult, because of the variety of similar, but not identical diagrams. It is difficult to see how the diagrams depend on each other.

2.2 Improvements

2.2.1 *Step 1: Introducing behavior models*

- As a first step we introduced a higher level of abstraction by writing models (as shown in figure 2). These models are realized as behavior diagrams (a kind of state diagrams). These diagrams describe the system under test from the user's point of view.

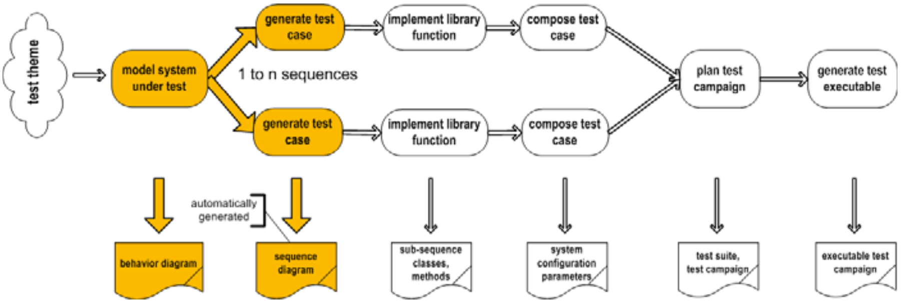


Fig. 2: Introducing usage models and automatic test case generation

Every state or transition is then associated with the sub-sequence diagrams already mentioned in section 2.1.1. As before, the same sequence diagram can be assigned several times, e.g. to different transitions. The diagram itself is only implemented and maintained once in the test library.

In addition to the re-use of sequence diagrams from the test libraries, we can also re-use behavior diagrams. For example, diagrams that describe error situations on the CAN bus can be included whenever messages are exchanged with the bus, thus enhancing the test coverage.

The complete set of sequence diagrams that correspond to the possible paths through the model is then generated automatically from the behavior diagram using the test case generator .getmore. The major advantages of the models are, that inter-dependencies between diagrams are visualized and that strategic aspects can be applied at test case generation. Moreover, the additional automatism increases the efficiency of the process and decreases the effort spent on maintaining the test cases.

2.2.2 *Step 2: Including test management information*

The first step has been successfully performed by sepp.med and Audi AG. The next step that is currently undertaken is to introduce additional information in the model. According to the philosophy of model-centric testing, all information relevant for the test should be contained in the model. This includes test data and test management information.

In this case, test data are the system configuration parameters that are defined when the test case is "composed". These can be parameter values or specific system behavior depending of the product variant.

The selection of test cases included in the test campaign depends on the test management information. For example, some sequences can be excluded from the test due to the fact, that the product variant under test does not support the tested functionality. Also, some transitions are of lower priority than others, possibly because they are not critical or maybe they have been tested thoroughly before and not been modified since.

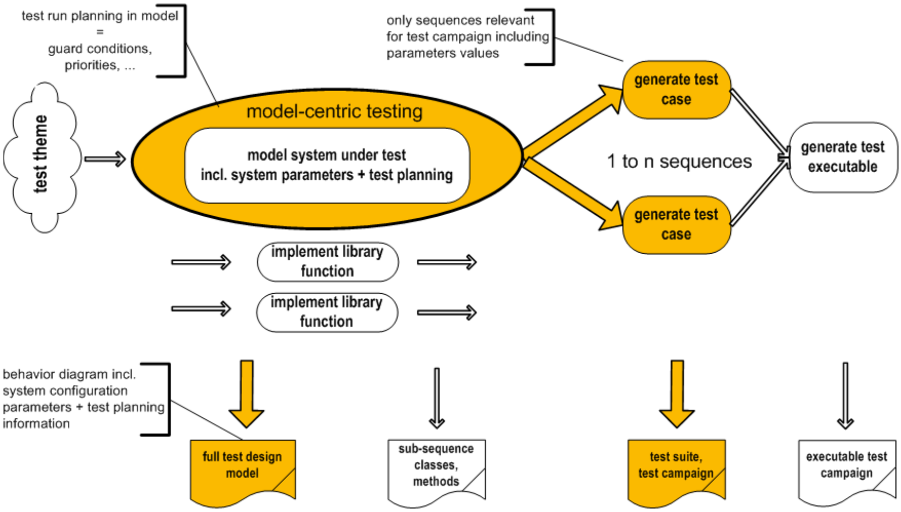


Fig. 3: Introducing test data and test management information

In figure 3 we show how model-centric test influences the workflow. Test data are included in the model. The programmers only implement the atomic code bodies. Modeling and programming activities are separated properly.

Once the test design model has been completed and checked, the complete sequence charts are generated automatically using .getmore. Please note the difference to figure 2, where the test cases correspond to generic test specifications. With the model-centric approach, only test cases that are relevant for the test campaign are generated. The planning of the test campaign was done in the model.

Afterwards, the executable test campaign is generated, executed and results are analyzed as before.

2.3 Benefits

The benefits of combining EXAM with model-centric testing are various. The additional behavior diagram introduced a higher level of abstraction. It becomes easier to understand the relationships and to discuss them with other stakeholders. In fact, the improved communication due to the graphical presentation is one of the major advantages of models in general. Models are easier to understand, which means that discussions get rapidly to the point and are less time consuming. Specification errors can be identified during the discussions and the overall quality of tests is improved.

The model-centric approach also introduces a high degree of systematic to the test. From the model it is clear which test cases exist and whether they are relevant for the test campaign or not. The test coverage can be defined and measured. In other words, we know what has been taken into account and why. With the process described in figure 1 this knowledge is merely contained in the tester's mind. With the model, it becomes understandable for others and, thus, reproducible.

We also obtained re-usability at a higher level. Today, the test EXAM libraries also contains sub-diagrams that are already fully implemented. This considerably reduces the effort for developing new tests and facilitates maintenance.

One very important benefit concerns cost-effectiveness. We did a direct comparison of the classical approach (as shown in figure 1) and the model-centric approach (figure 2). Tests for a selected functionality were developed by a test designer at Audi using the classical EXAM method. In parallel, sepp.med wrote the behavior diagram and generated test cases with .getmore. A comparison of the time spent showed that both projects took the same time. Thus, introducing the model-centric approach did not generate any additional effort. It is interesting to note, that sepp.med was not familiar with the system under test. With the same initial effort we obtain all advantages of using models, especially when it comes to maintenance. Models are far easier to maintain!

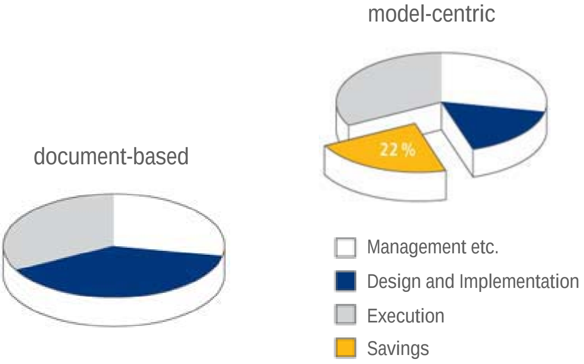


Fig. 4: Quantitative comparison of two projects (effort of test activities only)

In a different context we also performed a quantitative comparison of two projects (see figure 4) we did with customers from the medical device industry. The projects have been chosen in a way that they are comparable with respect to overall effort (approx. 7 person-years), amount of new requirements, duration, environment and complexity. In both projects, system tests were designed, implemented and executed. In one case, we used model-centric test design, whereas the other project was conducted with document-based tests. Test management (including team meetings and controlling) and test execution took the same amount of time, i.e. approximately one third of the overall effort of the document-based project. This is not very surprising. For these activities, it does not make any difference, how the test cases have been designed. The essential advantage can be observed during test design and implementation, were the model-centric test design saved more than 50% effort with respect to the document-based approach. Note that these savings have been obtained even without using automatic test case generation.

3 Summary

We combined the concept of model-centric testing with the EXAM test system at Audi AG. The combination resulted in considerable improvements especially concerning the documentation of the way, how the test idea was obtained from the analysis of the system under test, and of the documentation, which test cases were selected for a test run and which test cases were excluded.

To measure the cost effectiveness we had the rare opportunity to perform a direct comparison of the two approaches. It turned out that the model-centric approach did not generate additional initial costs for test design, but provides all advantages of using models.

Audi gave us excellent feedback and the next step is already in work. Further activities concerning the parameterization of test results are planned.

4 Acknowledgement

We wish to thank Mr. Gerhard Kiffe from Audi AG for his valuable input to this paper.

5 Literature

[1] interview with Frank Köhler and Gerhard Kiffe, EXAM project leaders at Volkswagen AG and Audi AG,

http://www.micronova.de/testautomation/referenzen/89-interview-mit-vw-und-audi.html

[2] G. Götz, A. Metzger, "Challenges and Solutions for Test Design and Management for Next Generation Medical IT Testbed", in "Setting Quality Standards", Proceedings of the CONQUEST 2008, dpunkt.verlag Heidelberg, 2008

[3] www.modellzentrierterTest.de

6 Author CVs

Dr. Anne Kramer

Anne Kramer was born in 1967 in Bremen (Germany). She studied Physics at the University of Hamburg where she obtained her diploma in 1992. In 1995 she graduated at the Université Joseph Fourier in Grenoble (France). Immediately afterwards she started working for Schlumberger Systems in Paris – first as software developer for smart card tools, then as project manager for point of sales terminals.

In 2001 she joined sepp.med (near Erlangen, Germany), a service provider specialized on IT solutions with integrated quality assurance in complex, safety-relevant domains. Currently, Anne Kramer is working as project manager and process consultant.

Make Test Process Assessment similar to Software Process Assessment – The Spice 4 Test approach

Monique Blaschke
Michael Philipp
Tomas Schweigert
SQS AG, Stollwerckstr. 11, 51149 Köln
monique.blaschke@sqs.de; michael.philipp@sqs.de; tomas.schweigert@sqs.de

Abstract

The paper presents an ISO/IEC 15504 II conformant test assessment approach. It argues the need for such a method by reviewing the common approaches of ISTQB, TMM and TPI/TMAP and it outlines a first version of Spice 4 Test. The paper shows how ISO/IEC 15504 V was used as a starting point and it describes the actual model.

Introduction

SQS Group is a company with approximately 1400 employees and the leading company for providing testing and quality management services since 1982. To ensure the quality of work it was necessary to deal with process models. Based on decades of project experience, as well as using the knowledge form SPI-services a specific Test Process Assessment model was developed and internally used. This test process model was aligned to the requirements of ISO 15504 TR 1998. Due to the changes from ISO 15504 TR 1998 to ISO IEC 15504 it was necessary to rework the process model. A core team with experienced test- and quality manager was formed and supported by leading process assessment experts of SQS. The authors of this paper are part of this core team. After a period of evaluation of potential process or process reference models the team decided to stay with the ISO 15504 using Part 5 as starting point for deriving a specific model to assess test processes.

The standard ISO/IEC 15504 was designed for being enhanced by developing specific Process reference models (PRM) and Process assessment models (PAM) [1]. In the meantime specific models are available. The best known is the automotive SPICE® developed by the user group in coordination with the automotive domain. Additional parts are published as Technical reports and provides an exemplar system life cycle process assessment model (Part 6) and for assessing organizational maturity (Part 7).

In the test business there are 2 major assessment methods in the market: TPI/TMAP and TMM. Besides there exists different schemes for the education of test professionals like ISTQB, which is recognized as a de facto standard at least in Germany. The ISTQB education scheme implies an own process model that is used e.g. for organizing the test from a test management perspective. Because of this our team decided to judge the ISTQB approach as a topic for the evaluation and as a potential source for test processes. The first question to answer was: Is one or more of these models compliant to ISO/IEC 15504 Part 2, and if not: how should a compliant model look like?

The conformance requirements of ISO/IEC 15504 II

The PAM shall declare

- the selected PRM(s)

- the selected processes taken from the PRM and the
- capability levels taken from the measurement framework.

It is also required, that the Model describes the mapping between the model and

- the Process Reference Model
- the Measurement Framework

As long as a model for assessing tests will follow these given structure the conformance to ISO/IEC 15504 Part 2 is assured. The SPICE 4 TEST approach is designed to fulfill these essential preconditions.

The currently available Test Process Models

There are 3 major models available on the market:

- ISTQB
- TPI®/TMAP®
- TMM®

lets have a look if these models fulfil the requirements of ISO/IEC 15504 II

ISTQB

The International Software Testing Qualifications Board (ISTQB) provides a set of syllabi for the qualification of test people (e.g. foundation level, advanced level: functional tester or test manager and expert level certified test process improver). [2] [3] [4] [5] Even tough we knew that the ISTQB does not claim to provide a process reference model, the team decided to include the model in the evaluation. The ISTQB syllabus provides a fundamental test process: planning and control; analysis and design; implementation and execution; evaluating exit criteria and reporting; test closure activities. It also contains a glossary [8] [9]. The description of processes is heterogeneous. Sometimes a process is described with its purpose, but no explicit description of outcomes is available. Based on this the fundamental test process of ISTQB does not meet the conformance requirements of ISO/IEC 15504 Part 2. Never the less the content of this model was considered as a useful input by our team.

TPI®/TMAP®

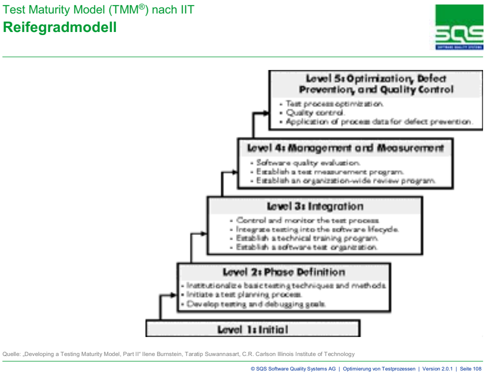
TPI®/TMAP® is the test process assessment and improvement method of a company named SOGETI. The assessment is based on a questionnaire that covers the needs of software testing. The approach uses a 2 step maturity model. Each check point can be fulfilled in up to 4 levels (A..D) which require the fulfilment of different aspects of the check point.. These levels are mapped to 3 general maturity levels (controlled, efficient, optimising). There is no mapping to the capability levels and process attributes of ISO/IEC 15504 available in the model [7] [10] (This result is based on the 1st version of TPI. Since the development of this paper a new version was published, it might be that an analysis of this version will lead to another conclusion). Result: TPI®/TMAP® (1st version) does not meet the conformance requirements of ISO/IEC 15504 II

Key Area	Reached TPI Matrix													
	0	Controlled					Efficient					Optimizing		
1 – Test Strategy	🟩	A					B				C		D	
2 – Life Cycle Model	🟩	A			B									
3 – Moment of Involvement	🟩		A				B				C		D	
4 – Estimating and Planning	🟩			A							B			
5 – Test Design Techniques	🟩	A		B						C				
6 – Static Test Techniques	🟩				A		B							
7 – Metrics	🟩					A			B			C		D
8 – Test Automation	🟩			A				B			C			
9 – Test Environment	🟩			A				B						C
10 – Office and Laboratory Environment	🟩			A										
11 – Commitment and Motivation	🟩	A				B						C		
12 – Test Functions and Training	🟩			A			B			C				
13 – Scope of Methodology	🟩				A		B				C			D
14 – Communication	🟩		A		B							C		
15 – Reporting	🟩	A			B		C					D		
16 – Defect Management	🟩	A				B		C						
17 – Testware Management	🟩		A			B				C				
18 – Test Process Management	🟩	A		B								C		
19 – Evaluation	🟩		A				B			C				
20 – Low-Level Testing	🟩				A		B		C					
21 – Integration	🟩			A			B				C			

Fig. 1 The improvement strategy table of TPI® (Example Automotive TPI®)

TMM(sm)/TMMi®

TMM(sm) was initially developed from the Illinois Institute of Technology and is now maintained as TMMi® by the TMMi foundation. The objective of this initiative was to use the CMM®/CMMI® approach for test process assessment and improvement. The current published model is based on the staged approach of CMMi® that means processes are directly linked to maturity levels. Different to CMMI®, TMMi® has no continuous representation [6]. A continuous model allows to define the capability level of each process and to deliver a capability profile. Contrary to the staged model a continuous model has a chance to meet the conformance requirements of ISO/IEC 15504 II. Result: TMMi does not meet the compliance requirements of ISO/IEC 15504 II.



Test Maturity Model (TMM®) nach IIT

Reifegradmodell

Level 5: Optimization, Defect Prevention, and Quality Control
- Test process optimization.
- Quality control.
- Application of process data for defect prevention.

Level 4: Management and Measurement
- Software quality evaluation.
- Establish a test measurement program.
- Establish an organization-wide review program.

Level 3: Integration
- Control and monitor the test process
- Integrate testing into the software lifecycle.
- Establish a technical training program.
- Establish a software test organization.

Level 2: Phase Definition
- Institutionalize basic testing techniques and methods.
- Initiate a test planning process.
- Develop testing and debugging goals.

Level 1: Initial

Quelle: „Developing a Testing Maturity Model, Part II" Ilene Burnstein, Taratip Suwannasart, C.R. Carlson Illinois Institute of Technology

© SQS Software Quality Systems AG | Optimierung von Testprozessen | Version 2.0.1 | Seite 108

Fig. 2 The Maturity Levels and Processes of TMMI

Intermediate Result

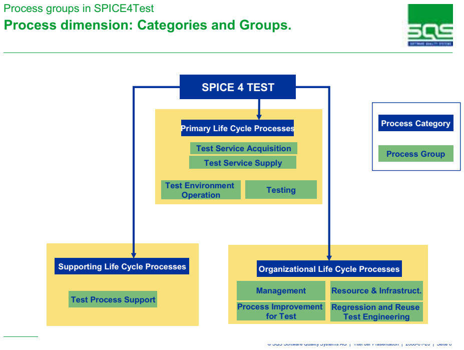
None of the current available test process assessment models meet the conformance requirements of ISO/IEC 15504 II

The SPICE 4 Test approach

Objective of the Test 4 Spice approach is, to deliver a PRM and a PAM that both meet the conformance requirements of ISO/IEC 15504 II and cover the processes necessary to effectively and efficiently assure the quality of software products.

The Basis: ISO/IEC 15504 V

The team developed SPICE 4 TEST with ISO/IEC 15504 V as starting point. This model is structured in process categories, process groups and processes. We decided to use the whole structure.



Process groups in SPICE4Test

Process dimension: Categories and Groups.

SQS
SPICE 4 TEST
Primary Life Cycle Processes
Test Service Acquisition
Test Service Supply
Test Environment Operation
Testing
Process Category
Process Group
Supporting Life Cycle Processes
Test Process Support
Organizational Life Cycle Processes
Management
Resource & Infrastruct.
Process Improvement for Test
Regression and Reuse Test Engineering

Fig. 1 The overall Structure of SPICE 4 Test

To transform the original model we used 3 methods

- Do nothing: a process is transferred 1:1 from ISO/IEC 15504 V to SPICE 4 Test Example: Project Management
- Replace: a process from the original model is replaced by a test process. Example: Domain Engineering was replaced by Regression Test Management
- Rename: Some process groups where renamed to show the focus of their processes. Example: Support was renamed to Test Process Support
- Insert: A new process was inserted Example: Test planning

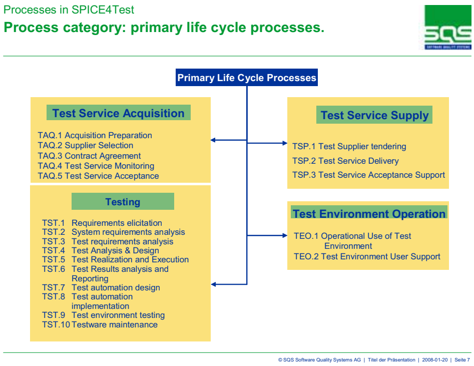
We did not change the process categories.

Sources

The team interviewed several colleagues from SQS and analysed the current available models (ISTQB, TMAP®/TPI®, TMMI®) and some literature to extract the common ideas about software testing processes.

The SPICE 4 TEST Model at a Glance

The following figures show the overall content structure of the model:



Processes in SPICE4Test

Process category: primary life cycle processes.

Primary Life Cycle Processes

Test Service Acquisition

TAQ.1 Acquisition Preparation
TAQ.2 Supplier Selection
TAQ.3 Contract Agreement
TAQ.4 Test Service Monitoring
TAQ.5 Test Service Acceptance

Testing

TST.1 Requirements elicitation
TST.2 System requirements analysis
TST.3 Test requirements analysis
TST.4 Test Analysis & Design
TST.5 Test Realization and Execution
TST.6 Test Results analysis and Reporting
TST.7 Test automation design
TST.8 Test automation implementation
TST.9 Test environment testing
TST.10 Testware maintenance

Test Service Supply

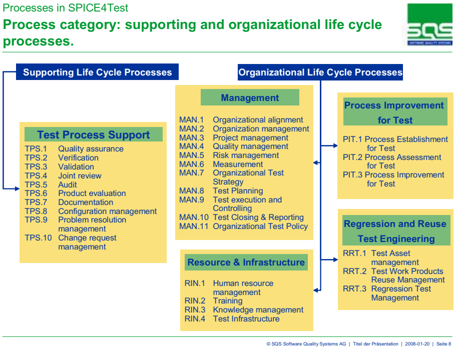
TSP.1 Test Supplier tendering
TSP.2 Test Service Delivery
TSP.3 Test Service Acceptance Support

Test Environment Operation

TEO.1 Operational Use of Test Environment
TEO.2 Test Environment User Support

© SQS Software Quality Systems AG | Titel der Präsentation | 2008-01-20 | Seite 7

Fig. 2 The primary life cycle processes of Test 4 SPICE



Processes in SPICE4Test

Process category: supporting and organizational life cycle processes.

Supporting Life Cycle Processes

Test Process Support

TPS.1 Quality assurance
TPS.2 Verification
TPS.3 Validation
TPS.4 Joint review
TPS.5 Audit
TPS.6 Product evaluation
TPS.7 Documentation
TPS.8 Configuration management
TPS.9 Problem resolution management
TPS.10 Change request management

Organizational Life Cycle Processes

Management

MAN.1 Organizational alignment
MAN.2 Organization management
MAN.3 Project management
MAN.4 Quality management
MAN.5 Risk management
MAN.6 Measurement
MAN.7 Organizational Test Strategy
MAN.8 Test Planning
MAN.9 Test execution and Controlling
MAN.10 Test Closing & Reporting
MAN.11 Organizational Test Policy

Resource & Infrastructure

RIN.1 Human resource management
RIN.2 Training
RIN.3 Knowledge management
RIN.4 Test Infrastructure

Process Improvement for Test

PIT.1 Process Establishment for Test
PIT.2 Process Assessment for Test
PIT.3 Process Improvement for Test

Regression and Reuse Test Engineering

RRT.1 Test Asset management
RRT.2 Test Work Products Reuse Management
RRT.3 Regression Test Management

© SQS Software Quality Systems AG | Titel der Präsentation | 2008-01-20 | Seite 8

Fig. 3 The Supporting and Organizational Life Cycle Processes of SPICE 4 TEST

Mapping from ISO/IEC 15504 V to SPICE 4 TEST

The following mappings show the original content of ISO/IEC 15504 V to SPICE 4 TEST.

Mapping by Process Groups

ISO/IEC 15504 V	SPICE 4 TEST
Acquisition	Test Service Acquisition
Supply	Test Service Supply
Operation	Test Environment Operation
Engineering	Test
Support	Test Process Support
Management	Management
Resource and Infrastructure	Resource and Infrastructure
Process Improvement	Process Improvement for Test

Reuse	Regression and Reuse Test Engineering

Mapping by Processes for the Acquisition Process Group

ISO/IEC 15504 V Acquisition	SPICE 4 TEST Test Service Acquisition (TAQ)
Acquisition preparation	TAQ.1 Acquisition Preparation
Supplier selection	TAQ.2 Supplier Selection
Contract agreement	TAQ.3 Contract Agreement
Supplier monitoring	TAQ.4 Test Service Monitoring
Customer acceptance	TAQ.5 Test Service Acceptance

Mapping by Processes for the Supply Process Group

ISO/IEC 15504 V Supply	SPICE 4 TEST Test Service Supply (TSP)
Supplier tendering	TSP.1 Test Supplier Tendering
Product release	TSP.2 Test Service Delivery
Product acceptance support	TSP.3 Test Service Acceptance Support

Mapping by Process for the Operation Process Group

ISO/IEC 15504 V Operation	SPICE 4 TEST Test Environment Operation (TEO)
Operational use	TEO.1 Operational Use of Test Environment
Customer support	TEO.2 Test Environment User Support

Mapping by Process for the Engineering Process Group

ISO/IEC 15504 V Engineering	SPICE 4 TEST Test (TST)
Requirements elicitation	TST.1 Requirements elicitation
System requirements analysis	TST.2 System requirements analysis
System architecture design	TST.3 Test requirements analysis
Software requirements analysis	TST.4 Test Analysis & Design
Software design	TST.5 Test Realization and Execution
Software construction	TST.6 Test Results analysis and Reporting
Software integration	TST.7 Test automation design
Software testing	TST.8 Test automation implementation
System integration	TST.9 Test environment testing
System testing	TST.10 Testware maintenance
Software installation	
Software and system maintenance	

As you can see this is not a mapping in the meaning that you can compare processes 1:1 but it helps you to understand the comparison between software engineering and software testing. Therefore both models use requirements elicitation as a starting point, because requirements are crucial for software engineers and for software testers. For the same reason, both models contain a maintenance process.

Mapping by Process for the Management Process Group

ISO/IEC 15504 V Management	SPICE 4 TEST Management (MAN)

ISO/IEC 15504 V Management	SPICE 4 TEST Management (MAN)
Organizational alignment	MAN.1 Organizational alignment
Organizational management	MAN.2 Organization management
Project management	MAN.3 Project management
Quality management	MAN.4 Quality management
Risk management	MAN.5 Risk management
Measurement	MAN.6 Measurement
	MAN.11 Organizational Test Strategy
	MAN.12 Test Planning
	MAN 13 Test Execution and Controlling
	MAN.14 Test Closing and Reporting
	MAN.15 Organisational Test Policy

The design of this process group reflects that on the one hand, there are standard processes in the management area and on the other hand there are specific processes to manage the test in the organisation and in the projects. Different to ISTQB Organisational Test Strategy and Organisational Test Policy are taken as processes (ISTQB: Work Products). Behind this more formal reason the design of the process group reflects the typical suffering of test teams:

- Poor project estimation
- Poor time planning
- Abuse of the planned test time as an undeclared time buffer for development activities
- Unrealistic goals
- Blaming the test team for slowing down the project speed.

The design of the process group allows to look at the test management as well as at the project management to see not only the symptom (test is late) but also if the symptom is caused by the test management or by the project management.

Mapping by Processes for the Resource & Infrastructure Process Group

ISO/IEC 15504 V Resource & Infrastructure	SPICE 4 TEST Resource & Infrastructure (RIN)
Human resource management	RIN.1 Human resource management
Training	RIN.2 Training
Knowledge management	RIN.3 Knowledge management
Infrastructure	RIN.11 Test Infrastructure

We changed Infrastructure to Test Infrastructure and gave a new ID to this process to make sure that assessors are looking for the right evidences.

Mapping by Processes for the Reuse Process Group

ISO/IEC 15504 V Reuse	SPICE 4 TEST Regression and Reuse Test Engineering (RRT)
Asset management	RRT.1 Test Asset management
Reuse program management	RRT.2 Test Work Products Reuse Management
Domain engineering	RRT.3 Regression Test Management

Mapping by Processes for the Process Improvement Process Group

ISO/IEC 15504 V Process Improvement	SPICE 4 TEST Process Improvement for Test (PIT)
Process establishment	PIT.1 Process Establishment for Test
Process assessment	PIT.2 Process Assessment for Test
Process improvement	PIT.3 Process Improvement for Test

We decided to rename the group and the processes to make sure, that only evidences relevant for the improvement of the test process are taken into account during an assessment.

Summary

If we consider ISO/IEC 15504 as an open standard for process assessment and improvement especially for the IT industry then this open standard should also be applied for test processes. The first version of this model shows, that this is achievable. As a benefit for the IT industry there is no longer a need to translate the results of proprietary models to the ISO/IEC 15504 measurement framework which saves money for training (one measurement framework fits all), data collection and analysis.

References

[1] ISO/IEC15504-5 Information Technology - Process Assessment - Part 5: An exemplar Process Assessment Model

[2] ISTQB® Certified Tester - Foundation Level: Grundlagen des Software-Testens (SQS Software Quality Systems AG, Version 2.10.0, Seminarunterlagen)

[3] ISTQB® Certified Tester - Advanced Level Test management (SQS Software Quality Systems AG, Version 1.4.1, Seminarunterlagen)

[4] Certified Tester - Foundation Level Syllabus (German Testing Board e.V. & Swiss Testing Board, deutschsprachige Ausgabe, Version 2007)

[5] Softwaretest - Lehrplan zum Aufbaukurs, ISTQB Certified Tester, Advanced Level (ASQF e. V., Version 1.2, Juli 2003)

[6] Test Maturity Model Integration (TMMi) (TMMi Foundation, Version 1.0, Februar 2008)

[7] TPI® Prüfkatalog (Übersicht TPI® Kontrollpunkte) (SOGETI)

[8] Standard glossary of terms used in Software Testing (‘Glossary Working Party’, International Software Testing Qualifications Board, Version 2.0, Dezember 2007)

[9] SQS-Glossar - Glossar gemäß ISTQB Certified Tester (SQS Software Quality Systems AG, August 2006)

[10] Martin Pol, Tim Koomen, Andreas Spillner, Management und Optimierung des Testprozesses, Heidelbeg, dpunkt, 2000

Authors CV

Monique Blaschke, Junior Consultant at SQS Software Quality Systems AG. Monique Blaschke studied from October 2003 to October 2006 Information Technologies at the Berufsakademie Stuttgart, a university for cooperative education and finished her study with the Bachelor of Science with honours in Information Technologies. After studying she worked two years for a great telecommunication company as technical project manager in different test projects for mobile communication, before she started in October 2008 to work for SQS. Apart from another test project for a telecommunication distributor, her major project there was the participation in the development of the test assessment approach SPICE 4 TEST.

Michael Philipp (Dipl-Kfm), Senior Consultant at SQS Software Quality Systems AG, Michael Philipp has a broad experience with software quality management, software testing and process improvement, staying with SQS from 1998. He is particularly characterized by experience in international projects (Testing, Improvement, IT-Life Cycle), Conducting assessments (ISO 15504) and as a trainer for SQS AG Training program. His actual Consulting focus is Quality Management, Test Management and Project Management. His special interest is the analysis of projects in crisis situations from the view of project management as well as the SPICE 4 TEST approach and risk management.

Tomas Schweigert (Ass. Jur.), Principal Consultant at SQS Software Quality Systems AG, Tomas Schweigert has a long experience with software quality management, software testing and process improvement, staying with SQS from 1991. He started his PI work with the BOOTSTRAP institute, working as a BOOTSTRAP lead assessor in several assessment projects. His special interest is the analysis of projects in crisis situations. Tomas Schweigert is now a SPICE principal assessor (INTACS scheme) his current research topics are the SPI Manager qualification and the test assessment approach SPICE 4 TEST.

A Model to Select Testable Use Cases: a Real Experience in a Financial Institution

Silva, Andrea Rodrigues[1]*, Rodrigues, Maikol Magalhães*[1]*, Pinheiro, Plácido Rogério*[1]*,*
Albuquerque, Adriano Bessa[1]*, Gonçalves, Francisca Márcia*[1]

[1]*University of Fortaleza (UNIFOR) – Graduate Program in Applied Computer Sciences*
Av. Washington Soares, 1321 - Bl J Sl 30 - 60.811-341 - Fortaleza – Ce – Brazil

Abstract. Nowadays, managers have to take increasingly complex decisions. This is due to several factors, including aggressive market competition, information overload and more demanding customers. Therefore, software organizations need to develop excellent and reliable products. This scenario has helped to increase the relevance of quality assurance activities, especially the testing discipline. However, sometimes time and resources are limited and not all tests can be executed, this leads organizations to decide what use cases should be tested to guarantee the time and budget project to be accomplished. Multiple criteria methodologies support decisions, considering many factors, not only professional experience. This paper presents a multiple criteria model to assist the selection of use cases that should be tested.

Keywords: Test, Decision and Analysis Resolution, Multiple Criteria Decision Analysis.

1. Introduction

Nowadays, the world's software industry increases because the software becomes part of many products and activities. According to Nollen [9], the worldwide software industry size was $1,045 billion in 2004. This scenario is motivating the software organizations to improve their products' quality, to meet the clients' needs, to reduce costs, to increase the productivity, to improve their time and cost predictability and to reduce the time to market.

Software testing is one of the disciplines that have the capability of providing assistance to improve the quality of an organization's products, because its goal is to evaluate how the product meets the clients' specified requirements through a controlled execution of the software. In some cases, when there is not enough time and resources to guarantee complete test coverage, software organizations should reduce their scope.

This work has as main objective the implementation of an approach based on a methodology that provides a structured support to multicriteria decisions. This methodology assists the process of deciding which use cases should be selected to be tested by removing subjective decisions in a structured way. The multicriteria methodology helps to generate knowledge about the decision context and thus increases confidence of those making decisions on the outcome results. This research was carried out to define and execute an approach to support software organizations at selecting use cases that should be tested. The approach was based on the model defined in [7].

2. Software Test

Software testing focuses on the product's quality and can not be considered elementary, because many factors can compromise the success of this activity's execution: (i) time limitations; (ii) resource limitations; (iii) lack of skilled professionals; (iv) insufficient knowledge of test procedures and techniques and adequate test planning; (v) subjectivity of requirement and test specifications; and (vi) increase of the systems' complexity.

Moreover, difficulties related to the test activity are also due to the great variety of combinations of input and output data and the large quantity of paths that make it infeasible to execute all possible test cases [2]. Test case is the definition of a specific condition to be tested. Its structure is based on input values, restrictions to its execution and expected results or behaviors [3]. Test cases are designed based on use cases, which represent interactions that users, external systems and hardware have with the software aiming to achieve an objective.

The amount of test cases executed is one of the main factors which may influence the cost of testing. Therefore it is fundamental to define a test scope considering acceptance criteria and business risks. Test criteria uphold the selection and evaluation of test cases aiming to increase the possibilities to provoke failures and to establish a high level of reliability on the products' errors correction [11]. The criteria may be classified as: (i) test coverage criterion; (ii) test cases adequacy criterion; and (iii) test cases generation criterion.

3. Multiple Criteria Decision Analysis

Decision-making should be exploited when deciding to execute or not some activities or to perform them applying some methods [7]. The multiple criteria decision analysis proposes to reduce the subjectivity on the decision-making. Nevertheless, the subjectivity will be always present, because the mathematically analyzed items are always results of human beings' opinions. These multiple criteria models allow the decision-maker to analyze possible consequences of each action to obtain the best understanding of the relationships between actions and their goals [6]. Objective criteria should be considered, as well as subjective criteria, even being generally disperse and diffuse in a decision context, but they are extremely important to assess actions.

The MACBETH methodology contemplates the understanding and learning of the problem content and is divided into three phases: structure, evaluation and recommendation [1]. The structure phase focuses on constructing a formal model, capable of being accepted by actors as a structure to represent and organize an entire group of evaluation criteria. It consists of analyzing a specific system and making potential alternatives of decision explicit. The evaluation phase produces matrixes of judgments and provides scales of cardinal value for every criterion. The tasks are implemented with the MACBETH methodology. In the recommendation phase, the results generated by MACBETH are analyzed using scales of values generated by the matrixes of judgments, which are composed of various actions that must be examined according to the decision-maker evaluation [7].

Structuring activities include: (i) definition of a family of fundamental points of view (FPV), (ii) construction of descriptors; and (iii) estimation of the impacts profiles of each action [1]. The construction of descriptors comprehends three stages: (i) description of each descriptor for each fundamental point of view; (ii) access impacts for each fundamental point of view, and (iii) analysis of impacts according to the fundamental points of view [10]. The descriptors are desired to: (i) turn operational the analysis of impacts of the options in a FPV, (ii) describe the impacts with respect to FPVs, (iii) improve the structure of the evaluation model, and (iv) verify the ordinal independence of the corresponding FPVs. The FPV becomes operational if there's a set of impact levels associated with it, defined by *Nj*, which should be sorted in descending order by the decision makers. Thus, they constitute a range of local preference, limited by the higher level N^*_j, that has more attractiveness and the lower level N_{*j}, of less attractiveness, should meet the following pre-ordering condition:

$$N_j^* > \ldots > N_{k+1,j} > N_{k,j} > N_{k-1,j} > \ldots > N_{*j} \,.$$

The main difference of MACBETH to other multiple criteria methods is that it requires only qualitative judgments of the difference between the elements' attractiveness aiming to assign values to the options for each criterion and to weigh up the criteria. MACBETH applies the concept of attractiveness to measure the potential actions' values. Therefore, when the decision-maker is demanded to judge the value of a potential action on a specific situation, he should think about his attraction to that action [4].

HIVIEW is a tool to evaluate models defined using multiple criteria methodologies with an aggregation function, like MACBETH. By using HIVIEW, the decision-maker defines, analyzes, evaluates and justifies his preferences, considering existent alternatives. This software facilitates a lot the analysis of complex problems, supporting the elaboration of the problem's structure, specifying the criteria used to choose alternatives and to assign weights to the criteria. The alternatives are evaluated by comparing these criteria and a preference value is assigned to each alternative’s criteria. Additionally, it’s possible to change judgments and to compare the obtained answers graphically, providing information to the decision-maker for reevaluation. If necessary the decision can be rectified.

4. Model to Select What Use Cases Should Be Tested and the Experience of Use

The proposed model is based on the model presented on [7] and is composed of generic steps, grouped by the phases of the Multiple Criteria Decision Aid (MCDA). These steps are described in Table 1.

Table 1. Model’s steps

Phase	Step	Description
Structure	1. Identify criteria	Identify criteria to be used on the use cases evaluation aiming to define their level of priority.
	2. Identify actors and their weights	Identify roles that will expose their point of view, considering also their roles on the decision-making process.
	3. Assign priorities to criteria	Each actor should assign a weight to all criteria, considering the full test process and not only a specific use case
	4. Execute a partial evaluation	After the execution of the steps listed above, it is necessary to standardize the three sets of values, putting them on the same base (base 1). The goal is to perform a partial evaluation correctly, without any bias. Then, for each actor the three variables should be multiplied, considering each criterion, thus obtaining a
	5. Calculate the general scores of the criteria	Calculate, for each actor, a score to each criteria of each use case.
Evaluation	6. Apply scores on MACBETH	Construct the matrix of judgments and obtain the cardinal value scales for each defined criteria.
Recommendation	7. Define the level of priorities of the use cases	Prioritize use cases which will be tested, given an analysis of the results obtained from the previous phases.

The organization where the experience of use was performed is a government institution with 270 professionals allocated on the Information Technology Area, working on projects that support the organization’s businesses.

In 2004, a test team was organized and being external to the projects and responsible for executing systemic tests. Nowadays, the company has high demands of time and resources which make it difficult to satisfy the desired testing coverage in all projects. Therefore, many projects reduce their testing scope to assure the delivery schedule. Priorities have to be applied to use cases and accordingly selected to decrease the testing scope. This is quite relative and varies according to the actors involved and to the criteria they judge relevant. The pilot project selected to apply the proposed approach was a project with a schedule restriction, because the organization agreed on a date with the workers’ union, resulting on a fine if it got delayed. The project’s life cycle was iterative/incremental and the model was applied on the project’s first iteration. The following use cases were part of the test cycle on which the model was applied: UC01_Execute_Sign_In_and_Sign_Out;UC04_ Search_Problems; UC05_Demand_Benefit_Permission; UC07_Search_ Demands_

Benefit_Permission;UC08_Approve_Demands_Benefit_Permission; UC10_Manage_Parameters; UC12_Excuse_Sign_in_and_Sign_out. Bellow, we present how the steps were performed.

In step "Identify criteria", we held a meeting with all actors involved at selecting the criteria, given the criteria listed on Table 2. The selected criteria are those highlighted. A criterion (c) is a tool to evaluate tests that are susceptible to automation in terms of a certain point of view (PV) or concern of the actors responsible for the analysis. The quantity of criteria (n) may vary for each project.

Table 2. List of criteria

Criteria	Reason	Question
Functionality [8]	The specific functions and properties of the product should satisfy the user.	Do the specific functions and properties of the product satisfy the user?
Usability [8]	The use case requires a strong interaction with the user and therefore must be usable.	Does the use case require a strong interaction with the user, being essential a high level of usability?
Security [8]	The use case requires a specific control, reducing the access to information.	Does the use case require a specific control to reduce the access to informations?
Repeatability [7]	The use case is being implemented on the first iterations and so will be tested many times.	Will the use case's test be repeated many times?
Complexity [7]	The use case has a large quantity of associated business rules or depends on complex calculations.	Does the use case have a lot of associated business rules or depend of complex calculations to function adequately?
User requirements [5]	The use case needs to be implemented because it will satisfy any contractual or legal demand or the organization may have financial loss.	Does the use case have to be implemented because it will satisfy any contractual or legal demand or it will prevent the organization against large financial loss?
Operational characteristics [5]	The use case is related to the functions which are frequently used.	Is the use case related to frequently used functions?

All stakeholders participated in a conference session to select the criteria from the list of criteria presented above, which should be considered to prioritize the use cases, as presented in Figure 1.

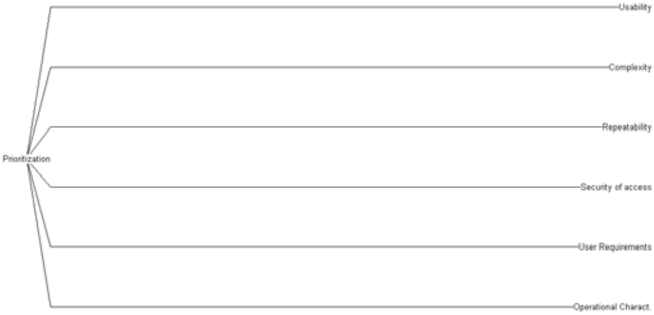


Fig. 1. Criteria selected for analysis

In step "Identify actors and their weights", the following actors were selected: (i) Project manager; (ii) Tests coordinator; (iii) Project's system analyst; and (iv) Project's test analyst. The actors answered the questions related to each criterion and the questionnaire applied to obtain the actors' weight, which was defined considering the role performed by the actor on the project e his knowledge of the business for which the software was developed. A questionnaire was elaborated to obtain the weight of each actor (weight of actor – WA), embracing actor's experience in activities related to tests; roles performed; participation in projects; training and participation in test conferences. Each item had a value and with the measurement of all items, the actor's weight was obtained. In step "Assign prioritization to the criteria", each actor (a) classified the criteria according to their relevance for the project's test process. In step "Execute a partial evaluation", we multiplied, for each use case, the values of the actors' point of view (PV), the actors' weights (WA) and the criterion's level of priorities, as can be seen on the formula below:

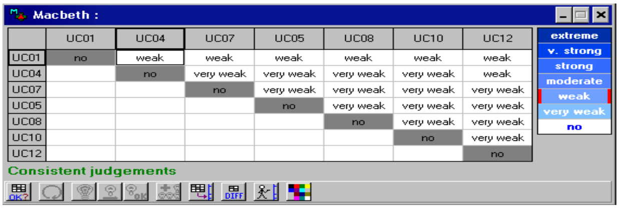
$$E_x\ (a, c) = [PV_x\ (a, c)] * [WA\ (a, c)] * [\text{Level of Prioritization } (a, c)]$$

It is important to emphasize that the obtained values of the actors' weights and level of priorities were equalized (on the same base – base 1), after they were informed so that a correct evaluation was possible without benefiting a value to the detriment of another. Finally, in step "Calculate the general scores of the criteria", we calculated the median to obtain the final score of the use cases for each criterion. The value of the median, calculated for each use case, will be used as a basis to prioritize. The following equation illustrates the median calculation.

$$Md(x,c) = \begin{cases} [E_{x,\ j/2} + E_{x,\ ((j/2)+1)}]/2, & \text{if } j \text{ is even,} \\ [E_{x,\ (j+1)/2}], & \text{otherwise.} \end{cases}$$

where x: represents the user case,
j: number of actors and
c: criterion

The evaluation phase was supported by the HIVIEW software. In step "Apply scores on MACBETH", we elaborated the matrixes of judgments in MACBETH for each criterion, having to calculate their subtraction of attractiveness. For this, the values of the median of the criterion are subtracted between the use cases (x_1 to x_k), considering the module of the obtained value. Figure 2 illustrates the matrix of judgments for the "Operational characteristics" criterion.



Macbeth :

	UC01	UC04	UC07	UC05	UC08	UC10	UC12
UC01	no	weak	weak	weak	weak	weak	weak
UC04		no	very weak	very weak	very weak	very weak	weak
UC07			no	very weak	very weak	very weak	very weak
UC05				no	very weak	very weak	very weak
UC08					no	very weak	very weak
UC10						no	very weak
UC12							no

Consistent judgements

Fig. 2. Matrix of assessments of the criterion "Operational Characteristics"

The model was applied at the beginning of the test cycle and we decided to test all use cases, aiming to analyze its adequacy and efficacy and to compare the obtained results. At the end of the test cycle, we calculated the percentage of errors, considering the quantity of use cases tested and the quantity of detected errors for each use case We could observe, considering the level of priorities assigned to the use cases obtained by the model execution, that the model assigned the highest level of priority to those use cases with a higher concentration of errors. However, the level of priority of UC07 was higher than those priorities of UC08, UC10 and UC12, but even so in UC07 were detected fewer errors.

5. Conclusion and Further Works

This work helped us to conclude that the restriction of time and resources is a real problem on many organizations and that an approach to support the decision to select use cases to be tested is very relevant. Besides, with the experience, we could see that the execution of the proposed model was satisfactory, allowing us to take the decision not using just our subjective experiences. By executing this approach we can learn three important lessons. First, we realize that one of our difficulties applying the model is the fact that the facilitator of this approach needs to know the multi criteria theme very well - it's too hard to find professionals with this profile in Brazilian companies. Second important lesson is that we notice how important is the facilitator to be exempted, to achieve trustful results. In other words, the facilitator should have a high level of independence against the project and test teams. And, finally, one of our biggest challenges applying the model is to get the actors together in a conference session. Most of the times they are very busy and do not have necessary commitment to have an effective participation in these sessions. The application of the MACBETH approach was adequate to model the problem but others Multi Criteria Decision Aiding (MCDA) methodologies can be used or, if necessary, we can define a combination of them to define the model.

As further works, it will be important to apply the model in other software projects and to define customized questionnaires for other projects' characteristics. The proposed approach can be used in other contexts, such as: the selection of processes improvements to be deployed, as it often is not possible to introduce all the improvements at once, given the difficulty of assessing and measuring the effectiveness of implemented improvements. The selection of systemic tests that can be automated is another scenario where we can apply the model. Criteria such as cost, availability of resources and repeatability of tests should be considered.

References

1. Bana & Costa, C. A.; Beinat, E. & Vickerman, R. (2001). Model-structuring and impact assessment: qualitative analysis of policy attractiveness. CEG-IST Working Paper n.25.
2. Bandeira, L. R. P.: Product Quality from Test Automation: a Case Study. Monograph to obtain the B.Sc in Computer Science Title, University of the Ceara State (2005)
3. Craig, R. D., Jaskiel, S. P.: Systematic Software Testing, Artech House Publishers, Boston (2002)
4. Cunha, J. A. O. Decisius: Um Processo de Apoio à Decisão e sua Aplicação na Definição de um Índice de Produtividade para Projetos de Software, 2008, Dissertation of M.Sc., CIN/UFPE, Recife, PE, Brazil.
5. Dustin, E.: Effective Software Testing: 50 Specific Ways to Improve Your Testing. - Addison-Wesley (2002).
6. Flament, M.: Multiple Criteria Glossary. Red Iberoamericana de Evaluación y Decisión Multicritério, Espanha, available at http://www.unesco.org.uy/redm/glosariom.htm (1999).
7. Gonçalves, F. M. G. S.; Donegan, P. M.; Castro, A. K.; Pinheiro, P. R.; Carvalho, A.; Belchior, A. D. : Multi-criteria Model for Selection of Automated System Tests. In: The IFIP International Conference on Research and Practical Issues of Enterprise Information Systems, 2006, Viena. IFIP International Federation for Information Processing (2006).
8. ISO/IEC 9126: Information Technology – Software Evaluation – Quality Characteristics and Guidelines for their use (2001)
9. Nollen, S.: Software Industry Performance in India and China, In: INDIA in the Emerging Global Order, eds. Amar K J Nayak and M G Jomon, Tata McGraw-Hill Publishing, Chapter 4 (2008)
10. Pinheiro, P. R.; SOUZA, G. G. C.; Castro, A. K. A.: Structuring of the Multicriteria Problem for the Production of Journal. Pesqui. Oper. 2008, vol.28, n.2, pp. 203-216. Digital Object Identifier 10.1590/S0101-74382008000200002.
11. Rocha, A. R. C., Maldonado, J. C., Weber, K. C. et al.: Software Quality – Theory and Practice, Prentice Hall, São Paulo (2001)

Softwares :
Hiview Software, Catalyze Ltd. www.catalyze.co.uk
M-Macbeth, www.m-macbeth.com

Test-Driven Automation: Adopting Test-First Development to Improve Automation Systems Engineering Processes

Dietmar Winkler Stefan Biffl Thomas Östreicher

Vienna University of Technology,
Favoritenstrasse 9/188, 1040 Vienna, Austria
{dietmar.winkler, stefan.biffl, thomas.oestreicher}@qse.ifs.tuwien.ac.at

Abstract

Software provides an increasing part of the added value of modern automation systems and thus becomes more complex. System requirements may change even late in the development process, lead to ad-hoc modifications of the product and require systematic (and automated) testing approaches. However, unit tests for automation software have to consider the interaction with hardware components, are often not systematically automated, and thus make defects during integration testing harder to find. Costly software integration makes the introduction of more flexible software processes that support the late change of requirements more risky. In this paper we introduce the concept of "Test-Driven Automation" (TDA), which adopts the successful idea of test-first development from business software development to the automation systems domain: develop test cases before the implementation and systematically automate unit tests to ensure sufficient testing on unit level to lower the cost and risk of systems integration. As foundation for TDA we present the characteristics of the design of a TDA software component, i.e., interfaces to (a) automation functions, (b) diagnosis functions to allow test observation, and (c) test functions for setting the component to defined states, e.g., to test behavior in error situations. We demonstrate in an industrial sorting application prototype how the TDA approach can make testing more efficient and provide diagnosis information for process analysis and improvement.

Keywords

Test-driven automation, automation systems, process support, testing, diagnosis.

1 Introduction

Software-intensive automation systems, like industrial manufacturing plants, need to become more flexible and robust to respond to changing business processes and business requirements. Functionality is increasingly realized in software components, which leads to an increased complexity of software components embedded within the hardware solution [17]. Engineers in the automation systems domain often have a non-software-engineering background and only limited knowledge on developing complex software-intensive systems. Late changing requirements and costly software and systems integration makes the introduction of more flexible software processes more risky [5][7]. Thus, effective and efficient software development methods and processes are necessary to support systems development and quality assurance across disciplines [8], i.e., software and automation systems.

Diagnosis and test are challenges in the automation systems domain to monitor and control current systems behavior during operation and maintenance [16] and respond to the current system status even before a failure occurs. In typical automation systems solutions we can observe software solutions that focus on functional requirements [18] and contain limited and often unsystematic diagnosis and test capabilities. Test and diagnosis aspects are scattered unsystematically in the code and hinders efficient re-validation and diagnosis. This ad-hoc approach to testing and diagnosis makes enhancements, refactoring and maintenance tasks more risky and expensive to validate [16]. Nevertheless, frequent and automated tests e.g., unit tests, can support engineers in finding defects early [2][9]. In contrast to business software development, unit tests in the automation domain have to consider interaction with the hardware [17]. These unit tests are often not systematically automated, and thus hinder efficient defect detection. A systematic separation of software aspects into functional behavior, testing and diagnosis aspects with well-defined communication and data exchange can increase product quality and support integration testing and maintenance for software-intensive automation systems. These characteristic aspects represent the foundation for "Test-Driven Automation" (TDA): interfaces to automation functions, diagnosis functions to allow test observation, and test functions for setting the component to defined states, e.g., to test systems behavior in error situations.

The concept of TDA adopts the successful idea of test-first development from business software development [2][9], e.g., administrative systems with databases, to help automation systems developers improve their development process. Test-first-development (TFD), an established method in business software development [2], focuses on quality assurance as an integral part of the development process and refers to the concept of early testing. This concept represents the foundation for automated unit-testing to lower the cost and risk of systems integration for iterative systems development [11]. Nevertheless, software processes are necessary to guide the engineers during the project. Software processes define sequences of steps along the product lifecycle and focus on specific needs of the project application context, like application domain and project attributes (e.g., size, complexity, and stability of requirements). In modern business software development a wide range of traditional (sequential) software processes, e.g., W-Model [1] and V-Modell, and flexible software processes, e.g., eXtreme Programming [2] and Scrum [3], support software engineers in developing high-quality products. The V-Modell XT[1], published in 2005, enables flexibility due to a modular process unit structure and provides a range of project execution strategies including agile approaches [6]. In the automation systems development industry the choice of systematic processes, e.g., GAMP [13], seems in practice limited to sequential processes, like waterfall models. To benefit from more flexible iterative approaches, automation systems engineers can learn from process approaches in business software development to handle the increasing complexity of software components in systems development.

The remainder of this paper is structured as follows. Section 2 introduces the concepts of TFD, Section 3 discusses requirements and challenges of the automation systems domain and identifies research issues for TDA. In Section 4 we introduce the TDA concept in context of a common engineering process. To illustrate how the novel TDA approach can increase testing efficiency and can provide diagnosis information for process analysis and improvement, we discuss an industrial sorting application prototype in Section 5. Finally, Section 6 summarizes lessons learned from the TDA prototype application and points out further research work.

[1] V-Modell XT resources available at http://www.v-modell-xt.de.

2 Test-First Development in Business Software Development

Traditional (sequential) software development approaches, e.g., waterfall models, place test case definition and test execution late in the development process after code construction. The identification of defects (caused by incomplete, wrong or ambiguous requirements) in late development phases can lead to (a) high effort to locate defects in the large application context, (b) high risk of defects that do not occur often, and (c) high rework effort to fix the problems [5].

Test-first development (TFD) [2] is a strategy to address these issues by shortening the cycles between test case definition and test execution [9]. Test cases are defined prior (or at least in parallel) to the implementation of a component. In general, the concept of TFD consists of 4 steps [2]: (1) Selection of a specific requirement and implementation of test cases to check the requirement for correctness (*Think*); (2) Test case execution. As there is yet no implementation of functionality, this test case must fail (*Red Test Result*). (3) Ongoing implementation of test-case related functionality and test-case execution until the test case is successful (*Green Test Result*). (4) Optimization of the implementation design without changing functionality and execution of test cases (*Refactor*). After finishing step 4 selection of the next batch of requirements. This approach can lead to a comprehensive understanding of the basic requirements and early defect recognition in case of unclear and incorrect requirements during test case generation. Additionally, TFD enables immediate feedback during component implementation and test because of frequent test runs in short iterations and thus represents the foundation for a continuous integration strategy [11].

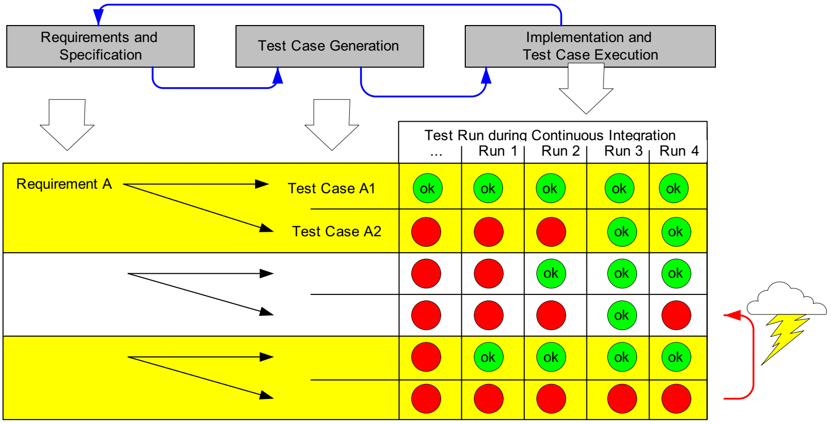

Requirements and Specification
Test Case Generation
Implementation and Test Case Execution
Test Run during Continuous Integration
...
Run 1
Run 2
Run 3
Run 4
Requirement A
Test Case A1
Test Case A2
ok

Figure 1: Concept of Test-First Development with several Test Runs.

Figure 1 illustrates the application of TFD in a typical project context in business software development. Test cases on business level (e.g., system and acceptance testing) can be derived from requirements (e.g., Requirement A maps to test cases A1 and A2). Frequent test runs during implementation provide immediate feedback on the current implementation task. Thus, implemented requirements and test case execution results lead to (a) successful test cases (marked green, requirement was already implemented correctly) and (b) unsuccessful test cases (marked red, implementation not finished or corrupted). TFD and continuous integration also helps identifying possible negative side-effects on other system parts (i.e., regression testing) which are not in the focus of the current implementation task (status switched from green to red without working actively on the affected code). For instance, the implementation of functions tested with test case C2 has a negative impact on the test case B2 result.

Frequent test runs in short iterations enable the observation of real project progress and deliver immediate feedback on the overall project status. Thus, automated testing is a pre-condition for continuous integration [11] including early and effective testing approaches.

3 Challenges and Research Issues in Automation Systems

In the automation systems domain we can observe a wide range of application types, e.g., production automation systems with focus on logistics and routing [19], embedded systems with limited resources, and real-time systems with time critical requirements [15]. In this paper we focus on applications of industry automation systems, e.g., assembly workshops to combine smaller parts into more complex products, to identify challenges for developing and testing software for automation systems [4]. Figure 2 illustrates the typical structure of an industrial automation system [12] on three layers: (a) business processes with dispatchers who turn customer contracts into work orders, (b) workshop operators for configuration and coordination of transport system and machines, and (c) layer of individual machines in the workshop with control systems from systems engineers and machine vendors. Plans and status reports link the layers: the current machine status can be used to reconfigure the current workshop (on workshop layer) and propagate status information to the business layer to reschedule work orders, if incidents on lower levels limit the effective workshop capacity [12].

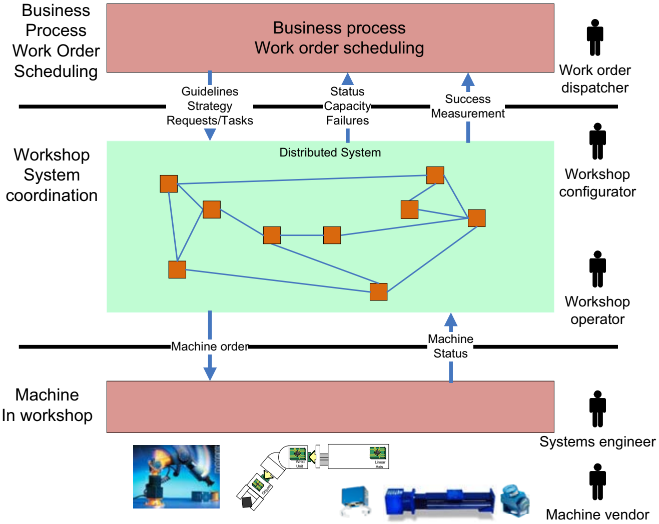


Figure 2: Levels of Automation Systems according to [12].

A typical systems engineering component in automation industry has to address (a) functional aspects to fulfill the required functional requirements [18], (b) diagnosis aspects to monitor and control systems behavior [16] and to predict upcoming maintenance needs, and (c) testing aspects to check systems behavior during systems development including tests cases on error conditions [16], e.g., regression testing after changes, and system tests during operation after completed maintenance tasks [17]. In traditional automation systems engineering the diagnosis and sometimes testing aspects are merged into the functional automation solution which hinders adapting efficient diagnosis and verification to changes during operation and maintenance.

Based on the Artemis[2] roadmap for automation systems research and discussions with industry partners in the Medeia[3] and logi.DIAG[4] research projects we derived the following needs and research issues:

[2] https://www.artemisia-association.org

[3] *Medeia: Model-Driven Embedded Systems Design Environment for the Industrial automation Sector*, http://www.medeia.eu/.

[4] *logi.DIAG: Test-Driven Automation in Systems Environments*, http://www.logidiag.at.

- *Separation of design aspects for automation functionality, testing, and diagnosis aspects.* Automation systems components typically include automation functionality, test-case functionality, and diagnosis functionality without a clear separation of these aspects, which makes the components unnecessarily hard to modify and validate [16][17]. Thus, we see the separation of these aspects as pre-condition for efficient systems development, operation, and maintenance to enable the selective evaluation of required attributes, e.g., test-coverage analysis during development. Test cases need to set the system in a certain state, e.g., stimulate an error state, and test the response of the system by using collected data from diagnosis. Thus, data derived from diagnosis functions can be used (a) for immediate feedback on the systems state (during development, operation, and maintenance), (b) for prediction of required maintenance tasks during operation, and (c) data can be used independently and combined as needed to get information on business level. *Thus, the first research issue is (a) how a TDA component can be designed and (b) how the individual aspects (automation, diagnosis, and test) can interact with each other efficiently.*
- *Efficient validation based on the test-first approach.* Efficient validation and re-validation after changes are key issues to reduce avoidable system downtime. Our observation in the automation systems domain showed a focus on functional requirements with limited and often unsystematic testing capabilities [18]. In the hierarchical design of automation systems we assume that TFD can be applied to components on all levels and in all process steps, e.g., on requirements level, architecture level, and component and module level. Process models, e.g. the V-Modell XT and the W-Model [1], can provide a framework for introducing systematic TDA. *The second research issue is (a) how the test-first approach can be realized in the production automation domain and (b) how the TDA concept can enable early testing on various levels during systems development.*

4 The Concept of Test-Driven Automation (TDA)

This section presents the concept of the TDA component structure and the interaction with its environment and introduces the concept of test-first development based on a suggested process approach for the automation systems domain.

4.1 Component Aspects in Test-Driven Automation

Bundling functional, diagnosis, and testing aspects in test-driven automation (TDA) components provide a strict separation of individual automation aspects. Interfaces enable an efficient communication in a hierarchical systems design. In common hierarchical automation systems, functional components and test cases are spread and mixed over the design of the systems. Diagnosis functions are typically add-ons without systematic integration within the systems design [16]. A strict separation of these components including defined interaction mechanisms are pre-conditions for efficient systems development, operation, and maintenance. Therefore, a test-driven automation component (TDA component) consists of these three aspects, (a) *automation aspect*, (b) *diagnosis aspect*, and (c) *testing aspect*. Figure 3 illustrates these aspects and Figure 5 provides the example of the prototype study.

The *automation aspect* implements the functional and logical behavior of the component and interacts via interfaces with other system components within a hierarchical systems design. The *diagnosis aspect* is separated from functionality but interacts with the automation aspect to get access to hardware signals. Instructions from automation functions are passed to sub-components via the diagnosis interface; measurements and results are received from sub-components and are (a) interpreted (diagnosis functionality) and (b) passed to the automation interface to respond to system overall results and signals. Additionally, diagnosis includes an interface to report measurement results to higher TDA components (e.g., for aggregation and reporting purposes). *Testing aspects* provide test functions for the TDA component, e.g., mocked results for the setup of test scenarios [14]. An important task of the test aspect is to set up error conditions (e.g., malfunctions of machines) for the unit test of diagnosis and logical functions without disturbing machine hardware components. Thus, these test functions can be applied for unit testing during development and during operation in case of sub-component exchange.

4.2 Test-First Development Approach with TDA Components

Based on the definition of the TDA component, including the three automation aspects, i.e., automation, diagnosis, and testing aspects, test-first development (TFD) can be applied in context of a software engineering process model. Our observations in automation systems industry reveals that typical engineering processes are similar to the waterfall approach. Thus, the V-Modell XT seems to be a promising basic model for systems automation development because of its flexibility and adaptability on various application domains [6]. Figure 3 presents the technical part of our suggested iterative model based on the V-Modell XT with respect to automation systems characteristics. Test-first development (TFD) is applicable on three levels (requirements, architecture & integration, and component) and enables early test case generation on every level.

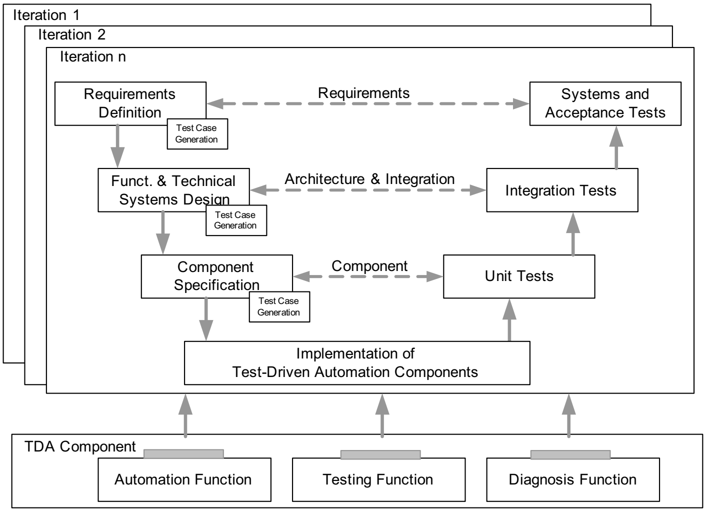


Figure 3: Concept of a TDA process based on the V-Model XT.

TFD includes the test case definition in the specification phases, i.e., requirements definition, functional & technical systems design, and component specification, of the process model and the execution of the test cases during and after implementation. Table 1 gives an overview on test levels of individual phases, deliverables generated during the entire phase, and relevant stakeholders.

Table 1: Test Levels and Deliverables for TDA according to the V-Model XT.

Phase	Deliverables	Test Level	Stakeholders
Requirements Definition	Use Cases	System / Acceptance Testing	Customer, Factory Setting
Functional and Technical Systems Design	Component diagrams, State-Charts	Architecture / Integration Testing	Engineering Team
Component Specification	State-Charts	Component Testing	Individual Engineer
Implementation of TDA Components	Function Blocks	Developer Testing	Individual Engineer

Test cases on requirements level address customer requirements and factory settings according to business goals and risks from business perspective. Note that test case definition on this layer provides a test framework for later development phases. Thus, underlying functionality must be mocked and simulated to enable successful test case execution [14]. This functionality can be provided by the testing aspect (i.e., setting the system in a certain state) and the diagnosis aspect (measuring results of the system) of the TDA component. On architecture and integration level test cases (and mocked individual components) address the engineering team and focus on components, interfaces and the interaction between components. Component diagrams and state charts are common models to describe components and interaction of these components. Component testing [9], based on a detailed state charts, addresses individual engineers and focuses on detailed functionality of the automation system. Function blocks and structured text are used for implementation purposes on the lowest and most detailed level of the process approach. This part must be considered during constructing individual components. In the context of this paper this phase is out of scope. Zhang *et al*. provide an approach for the specification and verification of applications based on function blocks. [20].

5 Prototype Study: Sorting Application

In this section we demonstrate the TDA approach with components in an industrial sorting application prototype, i.e., a typical machine in an assembly workshop as described in Section 3. The goal is to show how the TDA approach can make testing more efficient and provide diagnosis information for process analysis and improvement.

5.1 Sorting Application Description

The sorting application represents a typical setting in an assembly workshop within a production automation system. The main task of the sorting application is to recognize the type of an incoming clamp according to defined sorting criteria (e.g., derived from business goals) and sort it into the appropriate output box. The sorting application should manage to sort 6 clamps per minute. Figure 4a provides a schematic overview of the system.

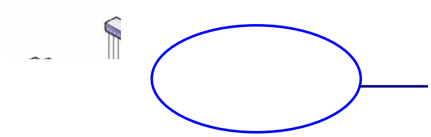


Figure 4: Sorting Application: (a) Workshop Layout and (b) Component Model based on Sünder *et al*. [18].

Figure 4b shows the component model of sorting application, a strictly hierarchical system as typical in industry automation system applications. To illustrate the TDA component and the interaction with its environment and to show the process of constructing models with respect to Model-Driven Testing [1] and generating test cases based on the test-first approach, we focus on a subset of components, i.e., the Handling Unit. The main task of the Handling Unit is to control two axes, i.e., the horizontal and vertical axis and the vacuum gripper to pick up the part for sorting purposes.

5.2 TDA Component of the Handling Unit

The TDA component encapsulates functional behavior, testing, and diagnosis functionality including interaction within the TDA component and providing/requesting interaction activities to other parts of the system using defined interfaces for automation, diagnosis, and testing. The component model of the Sorting Application (see Figure 4b) illustrates the interrelationship of the Handling Unit with other parts of the system: (a) *Loading Station* and (b) *Positioning Unit* and *Vacuum Gripper*. The Loading Station monitors and controls the Handling Unit via the functional interface, requests diagnosis data and initiate test cases (e.g., stimulates a defined state or a possible unreachable system state) via appropriate interfaces for diagnosis and testing. Note that the Loading Station represents another (higher-level) TDA component in the hierarchical systems structure. The Positioning Unit and the Vacuum Gripper (also – lower-level – TDA components) are controlled by the Handling Unit via functional interfaces (e.g., move to a certain position or grab a work product) and receives information whether the planned position is reached or the work product has been picked up via the appropriate diagnosis interfaces. Testing interfaces enable the Handling Unit to set the Positioning Unit and the Vacuum gripper in defined states, e.g., an error state.

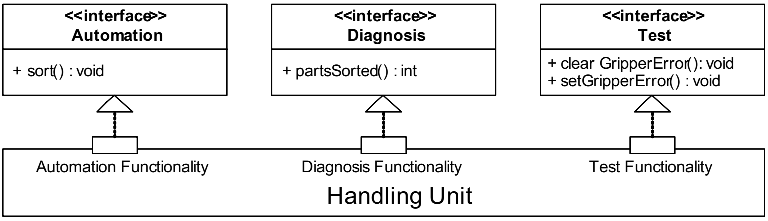


Figure 5: TDA Component Interface Model for a Handling Unit.

Figure 5 illustrates a simple component interface model of the Handling Unit. The component is triggered via the automation (functional) interface by the Loading Station `sort()` and returns the current state via the diagnosis interface `partsSorted()` after task completion. Error states are passed to the Vacuum gripper `setGripperError()` and `clearGripperError()` via testing interfaces to stimulate an error to test appropriate systems behavior of a supposed error of the gripper during development or maintenance. The strict hierarchical systems design of production automation systems enables TDA components to provide functional, diagnosis, and testing aspects to higher levels in the hierarchy. Thus, the TDA concept is, in principle, applicable on all levels of an automation system.

5.3 Test-First Development with Test-Driven Automation

The strict separation of functional, diagnosis, and testing functions in a hierarchical automation systems design based on TDA components is the foundation for successfully applying the test-first approach of the TDA concept. Following the suggested process steps (see Figure 3), test-first is applicable on three levels: (a) requirements definition, (b) functional & technical systems design, and (c) component specification.

Table 2: Test Cases derived from Requirements and Use-Cases.

No.	Desc.	Level	Type*	Pre-condition	Input	Expected Result	Post-condition
1	Sorting a Part	System	NC	Handling Unit in idle Position	Command to sort part	Handling Unit in idle Position and part sorted	Handling Unit in idle position
2	Through-put	System	NC	Handling Unit in idle Position	Command to sort part	Part has been sorted in less or equal than 10sec.	Handling Unit in idle position

Requirements definition. Requirements are success-critical issues in automation systems engineering as customer changes and modifications typically have a major impact on system integration and can cause a high amount of rework effort and cost [5][7]. UML use cases are appropriate approaches in business software development [1] to model requirements from user perspective and are applicable to the automation systems domain. The Handling Unit consists of a few use cases: (a) *sort part* (initiated by the Loading Station), (b) *Pick up part* and *Release part* (controlled by the Handling Unit), and (c) *Gripper Positioning* to get the arm moved to a certain position. Based on the basic requirements and the test-first approach of the TDA concept test cases can be derived directly from the use cases.

Table 2 presents selected regular/normal test cases (NC) on systems level. Note that test-cases should include normal test cases (NC), special test cases (SC) representing systems behavior in the border area of regular systems behavior, and error cases (EC) for error states of the system.

Functional and technical systems design and *component specification.* Based on use cases and the architecture of the hierarchical automation system (see Figure 4b) test cases can be derived on integration test level to test the interfaces and the interaction of the related TDA components prior to the detailed specification and the implementation (i.e., test-first approach). To enable automated testing based on test-first development, state charts can enable modeling the desired behavior of the system. In automation domain state charts are well-established and enable automated verification and validation [10]. Figure 7 illustrates the state chart of the Handling Unit including error states, which can be initiated via the testing interface and monitored via the diagnosis interface of the TDA component.

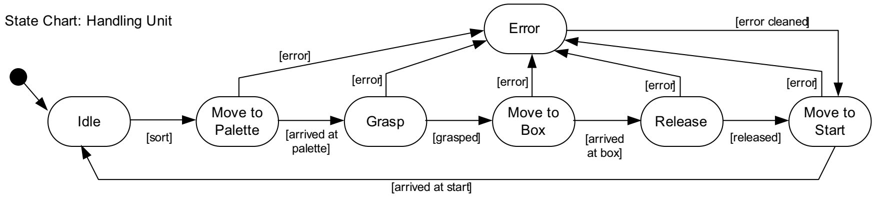


Figure 6: State Chart of the Handling Unit.

Table 3 illustrates selected test cases derived from the state chart, i.e., a regular test case and an error case, to demonstrate test case definition in TDA. Note that the gripper is in *idle* state (precondition) and should move to an *entire position*: Test case 1 describes a *regular test case* and test case 2 addresses an *error situation*, e.g., if one axis got stuck.

Table 3: Test Cases of the Handling Unit based on State Charts.

No.	Desc.	Level	Type*	Pre-condition	Input	Expected Result	Post-condition
1	Gripper move to Pos	Comp.	NC	Handling Unit idle	Sort part	Gripper moved to intended position	Gripper is in intended pos.
2	Axis got stuck	Comp	EC	Handling Unit in idle Position	Sort part; error after 3s	Positioning Unit reports an error; Handling Unit idle	Handling Unit in idle position

Testing automation systems also includes testing error states within a continuous integration strategy. In industry practice the definition of hardware error states (e.g., stuck of an axis) is an increasing challenge because (a) error states often must be initiated manually and are not feasible because of the availability of error states of machines and (b) required software components are not available during test time. Thus, these error states have to be simulated using mocking approaches to enable efficient testing and re-validation [14].

5.4 Mocking with State Charts

The application of TFD on various levels can require the simulation of the underlying functionality, if this functionality is not yet fully implemented. Thus, mocking of components is required to successfully apply TFD. A mocking component is a simple simulation of a subcomponent or external system which is required by a component to perform its behavior [14]. It should be as simple as possible, implementing only the behavior that is needed by the component under test. This facilitates test automation by enabling the testing of a component while its subcomponents are not yet fully implemented. Mocking also enables testing without deploying to the target hardware and the possibility of generating error states although an actual error is not present. Figure 7 shows a sample mockup in state chart notation of the Gripper Unit to simulate its basic behavior including error generation. Similar to test case generation mocking components are efficient modeling approaches for verification and validation purposes [14].

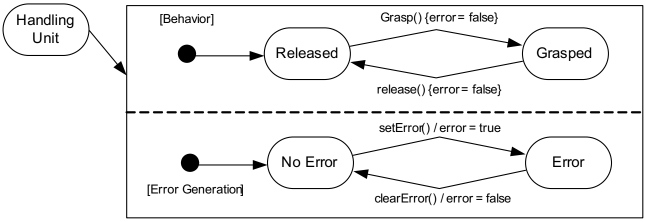


Figure 7: Mocking with Error Generation of the Gripper Unit.

The strict separation of individual software aspects and the interaction of individual TDA components enable a transparent systems design including testing and diagnosis capabilities. Based on the TDA concept TFD enables more efficient and systematic tests in comparison to the traditional automation systems approaches.

6 Summary and Further Work

The increasing need for flexibility of automation systems and the trend to shift functionality from hardware to software solutions leads to new challenges in the software development and to increased complexity of software. Up to now, code & fix approaches have been common practices in industry automation systems development. The increasing complexity requires more systematic software development approaches (methods and processes) for construction, refactoring, and verification and validation. In this paper we introduced the concept of “Test-Driven Automation” (TDA), which adopts the successful idea of test-first development from business software development to the automation systems domain and presented a novel TDA component including a strict separation of functional, testing, and diagnosis aspects to support more efficient testing on various levels of development.

Applying the presented concepts on a pilot sorting application we derived a set of lessons learned: *Packaging automation, testing and diagnosis aspects* in Test-Driven Automation (TDA) components provide strictly separated functions including well-defined communication over interfaces. Test functionality can put automation components into states for testing purposes that the automation logic would not reach with regular input, e.g., testing correct reaction on failure states. Diagnosis functionality enables measurement on the current system status and provides information for independently analysis of test and machine status and can support engineers during development, operation and maintenance, e.g., monitoring of systems behavior during operation and external data analysis for improvement purposes. *Test-first development* based on models can help to foster early test-case generation and increase the understanding of systems behavior and lead to higher product quality. TFD enables automated and frequent test case execution, support continuous integration and provides a framework for fast and efficient re-validation after changes in hardware and software components. Developing TDA components with the test-first approach lead to *an iterative development process* and provide well-defined and flexible framework for project execution.

Future work is (a) to refine the process approach with emphasis on automation systems development based on the V-Modell XT including domain-specific process tailoring and (b) to investigate the scalability of the TDA concept by addressing a larger pilot application with industry partners, with particular emphasis on data collection to compare the effectiveness of traditional testing in automation systems engineering and testing following the TDA concept.

Acknowledgements

We want to thank our partners from academia and industry in the Medeia and logi.DIAG projects for their valuable discussions and feedback. Parts of this work were funded by the Austrian Research Funding Agency (FFG) grant logi.DIAG (Bridge7-196929).

Literature

[1] Baker P., Dai Z.R., Grabowski J.: Model-Driven Testing: Using the UML Testing Profile, Springer, 2007.

[2] Beck K., Andres C.: Extreme Programming Explained - Embrace Change, Addison-Wesley, 2004.

[3] Beedle M., Schwaber K.: Agile Software Development with Scrum, Prentice Hall, 2008.

[4] Biffl S., Schatten A., Zoitl A.: Integration of Heterogeneous Engineering Environments for the Automation Systems Lifecycle, accepted for publication in Proc. IEEE Industrial Informatics Conf., 2009.

[5] Biffl S., Aurum A., Boehm B., Erdogmus H., Grünbacher P. (eds.): Value-Based Software Engineering, Springer Verlag, 2005.

[6] Biffl S., Winkler D., Höhn R., Wetzel H.: Software Process Improvement in Europe: Potential of the new V-Modell XT and Research Issues, in Journal Software Process: Improvement and Practice, Volume 11(3), pp.229-238, Wiley, 2006.

[7] Boehm B.: Software Engineering Economics, Prentice Hall, 1981.

[8] Chan K.K., Spedding T.A.: An integrated multi-dimensional process improvement methodology for manufacturing systems, Comput. Ind. Eng. 44(4): 673-693, 2003.

[9] Damm L.-O., Lundberg L.: Quality Impact of Introducting Component-Level Test Automation and Test-Driven Development, Proc. EuroSPI, 2007.

[10] Drusinksy D.: Modeling and Verification using UML Statecharts, Newnes, 2006.

[11] Duvall M.P., Matyas S., Glover A.: Continuous Integration: Improving Software Quality and Reducing Risk, Addison-Wesley, 2007.

[12] Lüder A., Peschke J., Reinelt D.: Possibilities and Limitations of the Application of Agent Systems in Control, Proc. Conf. On Concurrent Enterprising (ICE), Italy, 2006.

[13] GAMP 4: Good Automation and Manufacturing Practice – Guide for Validation of Automated Systems, 2001.

[14] Karlesky M., Williams G.: Mocking the Embedded World: Test-Driven Development, Continuous Integration, and Design Patterns, Proc. Emb. Systems Conf, CA, USA, 2007.

[15] Larsen K. G., Mikucionis M., Nielsen B., Skou A.: Testing real-time embedded software using UPPAAL-TRON: An industrial Case Study. In Proc. 5th ACM Int. Conf. on Embedded Softw.,. ACM Press, 2005.

[16] Nandi S., Toliyat H.A.: Condition Monitoring and Fault Diagnosis of Electrical Machines, Conf on Industry Applications, Phoenix, US, 1999.

[17] Schäfer W., Wehrheim, H.: The Challenges of Building Advanced Mechatronic Systems. In 2007 Future of Software Engineering (May 23 - 25, 2007). International Conference on Software Engineering. IEEE Computer Society, Washington, DC, pp. 72-84, 2007.

[18] Sünder C., Zoitl A, Dutzler C.: Functional Structure-Based modelling of Automation Systems, Jounal of Manufacturing Research, 1(4), pp405-420, 2007.

[19] Vyatkin V., Christensen J.H., Lastra J.L.M.: OOONEIDA: An Open, Object-Oriented Knowledge Economy for Intelligent Industrial Automation, IEEE Trans. on Industrial Informatics, vol. 1, pp. 4-17, 2005.

[20] Zhang W., Diedrich C., Halang W.A.: Specification and Verification of Applications Based on Function Blocks, Computer-Based Software Development for Embedded Systems (LNCS 3778), Springer Verlag Berlin Heidelberg, pp. 8-34, 2005.

Author CVs

Dietmar Winkler

Dietmar Winkler is researcher and lecturer at the Institute for Software Technology and Interactive Systems, Vienna University of Technology, Austria. He received an MS in Computer Science from Vienna University of Technology. In 2007 he worked as a guest researcher at Czech Technical University, Department of Cybernetics in Prague (CZ). In 2008 he received a scholarship by Siemens Austria and the Faculty of Informatics of TU Vienna for a 4 months research stay at the Fraunhofer Institute of Experimental Software Engineering in Kaiserslautern (DE). His research interests include software engineering and software processes, quality management and quality assurance, and empirical software engineering. Additionally, he is software engineering and quality management consultant in the automotive business domain and in public sector IT projects and project member of the research project “Test-Driven Automation with logi.DIAG”.

Stefan Biffl

Stefan Biffl is an associate professor at the Institute for Software Technology and Interactive Systems, Vienna University of Technology, Austria. He received MS degrees in Computer Science and Business Informatics from Vienna University of Technology (VUT) and University of Vienna, respectively, and a PhD degree from VUT. In recent years he worked as guest researcher at the Fraunhofer Institute for experimental Software Engineering (Kaiserslautern, Germany) and at Czech Technical University in Prague, Department of Cybernetics. His research interests are in the areas: Value-Based Software Engineering; Empirical Software Engineering, risk and quality management, and software process improvement. He is a member of the IFIP Technical Committee on Software Engineering (IFIP TC2) and principal researcher of the research project “Test-Driven Automation with logi.DIAG”.

Thomas Östreicher

Thomas Östreicher is researcher and project member of the research project “Test-Driven Automation with logi.DIAG”. His research interests are in the areas: modeling of software intensive systems, programming languages, and software process management.

Processes Implementation Sequences to Achieve the Maturity Level 2 in CMMI-ACQ

*Calvo-Manzano Jose A.[1], Cuevas Gonzalo[1], Gómez Gerzon[2], Mejia Jezreel[1], Muñoz Mirna[1],
San Feliu Tomás[1], Sánchez Angel[3]*
[1]Faculty of Computer Science, Polytechnic University of Madrid
{joseantonio.calvomanzano, gonzalo.cuevas, ma.munoz, tomas.sanfeliu}@upm.es,
jmejia@mpsei.fi.upm.es
[2]Universidad Autónoma de Tamaulipas, Unidad Reynosa Rodhe
ggomez@uat.edu.mx
[3]everis Foundation,
Angel.Sanchez@everis.com

Abstract

CMMI-ACQ is a model that provides guidance for organizations on 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. This paper addresses an implementation sequence among the processes areas and the generic goals at maturity level 2. To achieve this objective, Graphs Theory is used to represent the existing dependencies among CMMI-ACQ processes areas to find strongly connected cluster and cyclic processes areas. These clusters have helped to determine, using formal criteria, the implementation sequences of the acquisition processes areas at maturity level 2.

Keywords

CMMI-ACQ, Agreement Management, Acquisition Requirements Development, Measurement and Analysis, Project Monitoring and Control, Project Planning.

1 Introduction

The information technology (IT) outsourcing is the organizational decision to turn over part or all the process related to the development, maintenance or exploitation of systems [7]. The massive outsourcing trend, started in 1989 with Kodak's 10 year [12], and nowadays, the IT outsourcing is having a fast growth worldwide [7, 14], because the organizations have seen in IT outsourcing a way to achieve their strategic goals, reduce costs and improve customer satisfaction [8, 13].

However, the IT outsourcing can be danger if it is not planned, executed and managed carefully, according to recent studies [2], 20% to 25% of large IT acquisition projects fail within two years and 50% within five years. Mismanagement, the inability to articulate customer needs, poor requirements definition, inadequate supplier selection and contracting processes, insufficient technology selection procedures, and uncontrolled requirements changes are factors that contribute to project failure [2, 6].

The majority of project failures could be prevented if the acquirer learned how to properly prepare the contract, and manage the contracts and suppliers [2, 6, 9, 11].

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) [2, 5]. The Acquisition concept in the model is broader than the Outsourcing concept. While the Outsourcing concept focuses on the specific processes of the services supplier, the Acquisition concept or IT procurement covers the hardware and the Commercial off-the-shelf (COTS) solutions. Therefore, the CMMI-ACQ model offers a valid answer to the outsourcing processes.

The main challenge of using CMMI-ACQ is how to interpret its implementation to determine how to align these model components with processes, goals, and practices in a particular outsourcing environment.

The purpose of this paper is to formalize an implementation sequence of the CMMI-ACQ processes areas (PA) to achieve the maturity level 2 (ML2) in a company. To achieve this, process maps that represent the existing dependencies (ED) among PA and the Generic Goal 2 are necessary.

This paper is organized as follows. Section 2 gives a brief description on CMMI-ACQ, section 3 describes the procedure proposed in this paper, section 4 addresses an implementation sequence of the process areas, applying the previous procedure and finally section 5 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 [2, 5]:

- 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 processes improvement: 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 organizations to improve a set of related processes.

CMMI-ACQ contains 22 process areas: 16 Process areas are CMMI Model Foundation (CMF) and 6 Process areas focus on practices specific to acquisition addressing agreement management, acquisition requirements development, acquisition technical management, acquisition validation, acquisition verification, and solicitation and supplier agreement development [2, 5].

Table 1: Provides a list of CMMI-ACQ process areas and their associated abbreviations and maturity levels.

PA	Abbr.	ML
Agreement Management	AM	2
Acquisition Requirements Development	ARD	2
Configuration Management	CM	2
Measurement and Analysis	MA	2
Project Monitoring and Control	PMC	2
Project Planning	PP	2
Process and Product Quality Assurance	PPQA	2
Requirements Management	REQM	2
Solicitation and Supplier Agreement Development	SSAD	2
Acquisition Technical Management	ATM	3
Acquisition Validation	AVAL	3
Acquisition Verification	AVER	3
Decision Analysis and Resolution	DAR	3
Integrated Project Management	IPM	3
Organizational Process Definition	OPD	3
Organizational Process Focus	OPF	3
Organizational Training	OT	3
Risk Management	RSKM	3
Organizational Process Performance	OPP	4
Quantitative Project Management	QPM	4
Causal Analysis and Resolution	CAR	5
Organizational Innovation and Deployment	OID	5

It is very important to emphasize that the acquisition process (or purchase) of products and/or IT services is a continuous activity. The buyer must implement mechanisms to review the services or products delivered. The CMMI-ACQ model is not restricted because it covers both purchasing process patterns and the management of agreements and contracts established with suppliers.

3 OAP: Organization of the Acquisition Processes

The procedure described in this section, called OAP (Organization of the Acquisition Processes), and analyzes the ED among PA to propose an implementation sequence.

The main reason to elaborate this procedure was the need for a process roadmap that represents the ED among PA and helps us with the processes implementation. OAP is divided into three stages.

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

3.1 Identify Dependencies

The ED among PA was identified by reviewing the CMMI-ACQ official bibliography [2], through the 22 PA that the model includes. The analysis performed focused on the PA related to ML2.

3.1.1 Identify Dependencies

The model components were analyzed to detect references in order to identify the ED. The model components are the main architectural elements that compose a CMMI model. Some of the main elements of a CMMI model include specific practices, generic practices, specific goals, generic goals, process areas, capability levels, and maturity levels. The reference is an informative model component that points to additional or more detailed information in related process areas. An example of a reference found in the model components is the following paragraph:

> *"Refer to the Risk Management process area for mote information about to help determine whether a process area is satisfied".*

There are 4 components in the model components that contain the used reference:

- *Related Process Areas Section:* the related PA section lists the references to related PA and reflects the high level relationships among PA.
- *Specific Goals Section:* this section describes the unique characteristics that must be present to satisfy the PA.
- *Specific Practices Section:* this section contains the descriptions of activities that are considered important to achieve the associated specifics goals.
- *Subpractices Section:* this section provides guidance for interpreting and implementing a specific or generic practice.

A matrix of dependencies is elaborated from the ED found (Table 2). The rows represent the source processes and the columns represent the destination processes for the 9 PA of ML 2 (AM, ARD, CM, MA, PMC, PP, PPQA, REQM and SSAD). In cell Pij, the value 1 indicates there is a dependency between process i and process j (Pi∩Pj=1). The value 0 indicates there is no dependency between process i and process j(Pi∩Pj=0) [1, 4, 10,].

Table 2: Matrix of dependencies.

Destination		AM	ARD	CM	MA	PMC	PP	PPQA	REQM	SSAD	RSKM
Source	AM	0	0	1	1	1	1	1	0	0	0
	ARD	0	0	0	0	0	1	0	0	1	0
	MA	1	1	1	0	1	1	0	0	1	0
	PMC	1	0	1	0	0	1	0	0	1	1
	PP	1	1	1	0	1	0	0	0	1	1
	REQM	0	1	1	1	1	1	0	0	1	1
	SSAD	1	1	1	1	0	1	1	0	0	0
	ATM	1	1	0	0	1	1	1	0	0	0
	AVAL	1	1	0	0	0	0	0	0	0	0
	AVER	0	0	0	0	0	0	1	0	1	0
	DAR	0	0	0	0	0	0	0	0	1	0
	OPD	0	0	0	1	0	1	0	0	1	0
	OT	0	0	0	0	0	1	0	0	0	0
	RSKM	0	1	0	0	0	1	0	0	0	0
	OPP	0	0	0	0	0	1	0	0	0	0
	QPM	0	0	0	1	0	0	0	0	0	0

🟦 Processes Areas of the Maturity Level 2.

3.1.2 Represent Dependencies.

The dependencies shown in Table 2 are represented by the directed graph (digraph). Digraph D consists of a non-empty finite set V(D) of elements called vertices (or nodes) and a finite set A(D) of ordered pairs of distinct vertices called arcs (or edges). V(D) is the vertex set and A(D) is the arc set of D. D=(V, A) means that V and A are the vertex set and arc set of D, respectively [1, 10].

Fig. 1 and 2 show a generated digraph with the mathematical software tool from Table 2. The processes are represented by vertices and dependencies by arcs. Each vertex is labeled with the process acronym of the Table 1. The digraph shows the dependences from the 9 PA of the ML 2.

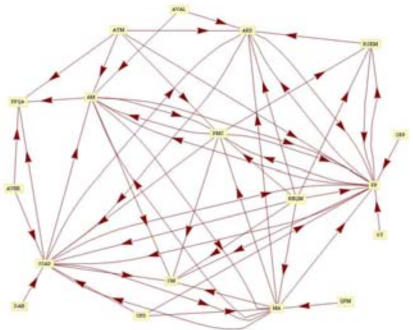


Figure 1: Total Dependencies among Process Areas of the Maturity Level 2.

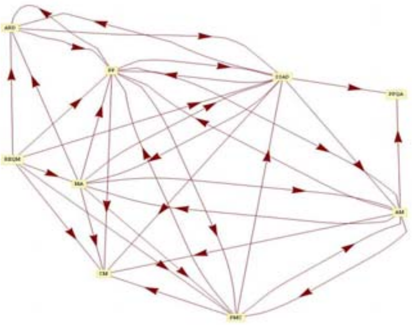


Figure 2: Dependencies among Process Areas of the Maturity Level 2.

3.2 Analyze dependencies

In the analysis stage the level of complexity was reduced considering only the dependencies between PA of the ML2 (see shaded section in Table 2), because ML2 is the first ML introduced in an organization. Figure 4 shows the digraph obtained.

3.2.1 Strongly connected components verify (SCC).

The arcs in the digraph shown in Fig. 4 were evaluated using a mathematical software tool [10] in order to check the SCCs. A subset D (V, A) of the digraph is strongly connected if each pair of vertices u and v (u≠v), has a path from u to v [1,10].

The following path which run through all the arcs and that is introduced as data in the Function of evaluation [10] StrongComponents[a] was built.

Input data:
Path={"AM"→"CM","AM"→"MA","AM"→"PMC","AM"→"PP","AM"→"PPQA","ARD"→"PP","ARD"→"SSAD","MA"→"AM","MA"→"ARD","MA"→"CM","MA"→"PMC","MA"→"PP","MA"→"SSAD","PMC"→"AM","PMC"→"CM","PMC"→"PP","PMC"→"SSAD","PP"→"AM","PP"→"ARD","PP"→"CM","PP"→"PMC","PP"→"SSAD","REQM"→"ARD","REQM"→"CM","REQM"→"MA","REQM"→"PMC","REQM"→"PP","REQM"→"SSAD","SSAD"→"AM","SSAD"→"ARD","SSAD"→"CM","SSAD"→"MA","SSAD"→"PP","SSAD"→"PPQA"};

After executing the mathematical software tool four SCC groups were obtained:

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

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

3.2.2 Generate combinations.

The sCr formula for combination is applied to SCC Group 3 in order to get all the different combinations of 3 processes, where s=6 (the number of elements of the selected SCC group) and r=3 (The minimal number of vertices in a cycle is 3 [1] and this is the reason for selecting groups of 3-processes). Results are shown in Table 3.

Table 3: Combination Results.

	Combinations				Combinations		
Group	**AP**	**AP**	**AP**	**Group**	**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

3.2.3. Cyclic groups.

The cyclical clusters are obtained (see Table 4) by applying the following criteria to each combination from Table 3:

- There is a cycle: A trail is a walk in which all arcs are distinct. If the vertices of W are distinct, W is a path. If the vertices v1, v2 ...vk-1 are distinct, k≥3 and v1=vk, W is a cycle [1].

Table 4. Cyclical Cluster.

Cluster	Processes		
D	PP	ARD	SSAD
F	PP	AM	PMC

3.3 Analyze dependencies including the Generic Goal 2

To achieve ML2, all generic practices assigned to Generic Goal (GG) 2 must be done. Generic goals are called generic because the same goal statement applies to multiple process areas [2]. For this reason, before beginning the implementation sequence stage, the analysis of ED between the GG2 generic practices and its related PA has been done.

The two stages of the OAP procedure have carried out for this analysis, including the ED of generic practices in the initial matrix (Table 2), obtaining the following results.

The Table 5 shows the updating of the values of shaded subtable in Table 2, according to the inclusion of the ED from GG2 generic practice.

Table 5: Matrix of dependencies with GG2.

		Destination								
		AM	**ARD**	**CM**	**MA**	**PMC**	**PP**	**PPQA**	**REQM**	**SSAD**
Source	**AM**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**
	ARD	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**0**	**1**
	CM	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
	MA	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**1**
	PMC	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**
	PP	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**
	PPQA	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
	REQM	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**1**
	SSAD	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**0**	**0**

3.3.1 Strongly connected components verify (SCC) including the Generic Goal 2.

The obtained values in the matrix shown in Table VI were evaluated using a mathematical software tool [10] in order to check the SCCs. After executing the mathematical software tool four SCC groups were obtained:

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

3.3.2 Generate combinations in the Generic Goal 2.

The sCr formula is applied to SCC Group 2 with the maximum number of elements in order to get all the different combinations of 3 processes, where s=7 (the number of elements of the selected SCC group) and r=3 (The minimal number of vertices in a cycle is 3). Applied the formula 35 combinations were obtained.

3.3.3 Cyclic groups including Generic Goal 2.

The cyclical clusters are obtained by applying the defined criteria in the section 3.2.3 to each combination. The Table 6 showed the combinations that cover the criteria.

Table 6: Cyclical Cluster with GG2.

Group	Combinations		
F	AM	MA	PMC
G	AM	MA	PP
J	AM	PMC	PP
W	ARD	PP	SSAD
Y	MA	PMC	PP
Z	MA	PMC	REQM
AB	MA	PP	REQM
AC	MA	PP	SSAD
AE	PMC	PP	REQM

3.4 Establish implementation sequences

According to the cyclical clusters obtained (Table 4 and 6), the implementation sequence cannot be implemented without the implementation of all processes that constitute the cyclical cluster.

In order to identify the processes implementation sequence, permutations for each cyclical cluster was generated from Table 4 and 6. A permutation was sorted from higher to lower number of source dependencies.

The implantation sequence begins with the basic level without including the objective generic (GG) 2. First, the PA implementation sequences of the Basic level are showed; second the continuations of the implantation sequences according the GG2 are showed.

1) ***The Basic Level Implementation of Generic Goal 1.***

The basic level only includes the GG1 to each PA. GG1 is equivalent to saying you achieving the specific goals of the PA.

The TSDN2 column in Table 7 (A) contains the Total Number of Dependencies that depends on the source process among PA of the ML2 and is defined as:

$$TSDN2 = \sum_{j=AM}^{SSAD} Pij. \qquad (1)$$

In order to identify the processes implementation sequence, permutations for each cyclical cluster has been generated from Table 4. A permutation is sorted by higher to lower number of source dependencies, according to the TSDN2 column in Table 7 (A).

Table 7: Source dependencies (A) and Permutation (B).

(A)

AP	TSDN2
AM	5
ARD	2
MA	6
PMC	4
PP	5
REQM	6
SSAD	6

(B)

Permutation	Implementation sequence		
	1º	2º	3º
F1	PP (5)	AM (5)	PMC (4)
F2	AM (5)	PP (5)	PMC (4)
D	SSAD (6)	PP (5)	ARD (2)

Table 7 (B) shows the implementation sequences probabilities. According to the TSDN2 column in Table 7 (A), there are two possibilities for the cyclical cluster F of the Table 4. The possibilities are permutations F1 and F2 of the Table 7 (B). In the case of cluster D, processes area already ordered.

2) The Basic Level Implementation of Generic Goal 2.

The GG2 describe the same goal statement applies to multiple process areas. The Table 8 (A) shows the Total Number of Dependencies that depends on the source process among PA and generics practices of the GG2. In order to identify the processes implementation sequence according GG2, permutations for each cyclical cluster has been generated from Table 6. A permutation is sorted by higher to lower number of source dependencies.

Table 8: Source dependencies with GG2 (A) and Permutation with GG2 (B).

(A)

PA	TSDN2
AM	5
ARD	4
CM	1
MA	8
PMC	8
PP	8
PPQA	1
REQM	7
SSAD	6

(B)

Permutation	Implementation sequence			Total Dependences by permutation
	1º	2º	3º	
Y	PP (8)	PMC(8)	MA (8)	24
Z1	PMC (8)	MA (8)	REQM (7)	23
Z2	MA (8)	PMC (8)	REQM (7)	23
AB1	PP (8)	MA (8)	REQM (7)	23
AB2	MA (8)	PP (8)	REQM (7)	23
AE1	PP (8)	PMC (8)	REQM (7)	23
AE2	PMC (8)	PP (8)	REQM (7)	23
AC1	PP (8)	MA (8)	SSAD (6)	22
AC2	MA (8)	PP (8)	SSAD (6)	22
F1	MA (8)	PMC (8)	AM (5)	21
F2	PMC (8)	MA (8)	AM (5)	21
G1	PP (8)	MA (8)	AM (5)	21
G2	MA (8)	PP (8)	AM (5)	21
J1	PP (8)	PMC (8)	AM (5)	21
J2	PMC (8)	PP (8)	AM (5)	21
W	PP (8)	SSAD (6)	ARD (4)	18

As in the implementation sequence of GG1 there are cyclical groups with more than one order of the implementation sequence in the GG2 (see Table 8 (B)) e.g groups Z=Z1,Z2, AB=AB1,AB2, AE=AE1,AE2, AC=AC1,AC2, F=F1,F2, G=G1,G2, y J=J1,J2 according to criteria TSDN2 of the Table 8 (A).

4 Proposal implementation sequence

The resulting groups the ED among PA of the ML2 and the ED of the GG2 inclusions were analyzed to give the proposal implementation sequence of (Table 7 (B) and Table 8 (B)).

The cyclical cluster F has two implementation sequences possibilities (F1 or F2, see Table 7 (B)). In the case of the cyclical cluster F, the implementation sequence would be permutations F1 or F2. According to business goals, the implementation sequence would be permutation F2 because Project Planning PA cannot be implemented without first updating the Agreement Management (AM) (see Table 9).

Table 9: Implementation sequence alternatives.

Permutation	Implementation sequence		
	1º	2º	3º
F2	AM (5)	PP (5)	PMC (4)
D	SSAD(6)	PP(5)	ARD(2)

Table 9 shows the two related implementation sequences alternatives according GG1, F2 and D permutation. Neither implementation sequences can be implemented without implementing all the processes that constitute the permutation.

- Permutation F2: this implementation sequence is selected when there is an Outsourcing Agreement. The implementation sequence begins with AM, then PP and finally PMC.
- Permutation D: this implementation sequence is selected when there is no Outsourcing Agreement. The implementation sequence begins with SSAD, then PP and finally ARD.

To achieve the ML2, the implementation sequence is defined according to the resulting cyclic groups of the ED inclusion analysis of the GG2 generic practices (Table 10). Selecting the cyclical groups which have the greatest weight (Column Total Dependencies by permutation of the Table 8 (B)) and that incorporate PA that are not reflected in the sequences of the GG1 of the ML2 (Table 9).

Table 10: Implementation sequence alternatives with GG2.

Permutation	1º	2º	3º	TD
Y	PP (8)	PMC(8)	MA (8)	24
Z1	PMC (8)	MA (8)	REQM (7)	23

Due to PP and PMC are already considerate in the previous cyclical groups (see Table 9), the implementation sequences number 3 is the Y cyclical group (Table 10), then MA and REQM PA are remaining to implement. The Fig. 3 shows the whole implementation sequences.

everis consultants, an IT outsourcing and business consulting firm with presence in Europe and Latin America makes the following considerations about the two implementation sequences alternatives.

- *In the case of Contract Management:* once the supplier agreement has been defined and awarded, it must be managed *(AM)* according to the defined SLA (Service Level Agreement). This management is supported by the project planning activities (PP) to provide the baseline and the minimum contract workload. In addition, it requires monitoring and control activities (PMC). It refers to permutation F2.
- *In the case of Defined Contract:* the Request for Proposal, Request for Quotation or agreements are defined to bid for the IT services outsourcing. In this step, the organizations decide which functions and what portions of the function they wish to outsource, the expected quality level and identify the potential Suppliers. This definition is supported by the project planning (PP) activities to know the baseline (The minimum workload to outsource). Then, organizations can develop outsourcing requirements (ARD). It refers to permutation D.

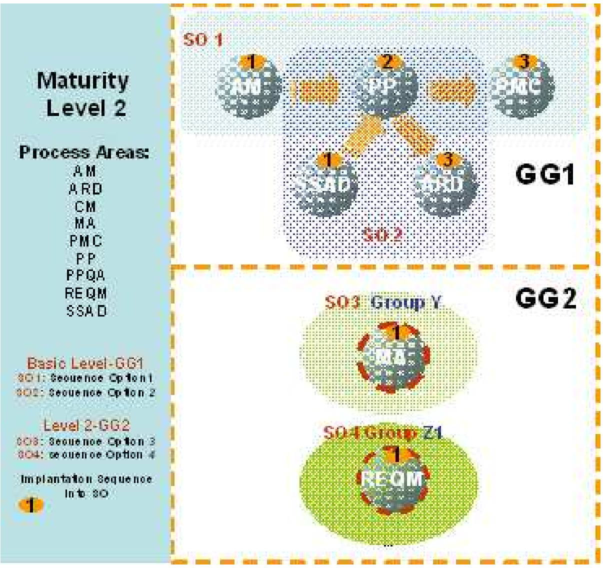


Figure 3: The whole ML2 implementation sequences proposal.

The organization knows how to launch a contract, to plan work to outsource, and to give the right requirements by completing the initial and essential steps. Besides, it is able to manage the contract through using the Service Level Agreement (SLA). SLA relies on the project planning and control information. This is the point where the organizations is able to formalize a Measurement and analysis practice that allows to know if the outsource process that feedback the outsource strategies (SSAD) is profitable or not. (Implementation Sequence Y: MA)

Finally, the organization must be able to adapt to the requirements and demand management to the outsource process (implementation sequence Z1: REQM). This adaptation is frequently call portfolio management.

5 Conclusions

The CMMI-ACQ official bibliography only shows the high level relationships among PA. Therefore, it is impossible to detect an implementation sequence. The goal of this paper was to identify and analyze the ED among the PA related to ML2 in order to propose an implementation sequence of these PA. Digraphs theory and formal criteria were used to propose the implementation sequences. The analysis procedure was done in the 2 levels that allow achieving the Maturity level 2.

First, it's propose the basic level implementation sequence that refers the Generic Goal 1 and, second it's complete the implementation sequences with the obtained sequences related to Generic Goal 2 generic practices.

The implementation sequence alternatives proposed apply when beginning the Outsourcing contract lifecycle or the organization already has an Outsourcing Contract in.

Finally, according to the considerations of everis consultants, it is concluded that the implementation sequence alternatives obtained by OAP shows a roadmap to implement CMMI-ACQ model in the ML2, which is largely compatible within the business practice. This traceability is an evidence of the adequacy of the proposed framework.

Acknowledgements. This work is sponsored by Endesa, everis Foundation, Sun Microsystems, and Polytechnic University of Madrid through the Research Group of Software Process Improvement for Spain and the Latin American Region.

6 Literature

1. Bang-Jensen., J., Gutin., G.: Digraphs : theory, algorithms and applications.: Springer-Verlag (2002)
2. Cmmi-Product-Team.: CMMI for Development Version 1.2., Software Engineering Institute (2006)
3. Dibbern, J., Goles, T., Hirschheim, R., Jayatilaka, B.: Information systems outsourcing: a survey and analysis of the literature. The DATABASE for Advances in Information Systems., 35., 6-102 (2004)
4. Diestel, R.: Graph Theory., New York., Springer-Verlag (2006)
5. Dodson, K. M., Hofmann, D. H. F., Ramani, G. S., Yedlin, D. K.: Adapting CMMI for Acquisition Organizations: A Preliminary Report., Software Engineering Institute (2006)
6. Gellings, C.: Outsourcing Relationships: The Contract as IT Governance Tool. 40th Conference on System Sciences. Hawaii International (2007)
7. Jens, D., Tim, G., Rudy, H., Bandula, J.: Information systems outsourcing: a survey and analysis of the literature. The DATABASE for Advances in Information Systems., 35., 6-102 (2004)
8. Mojsilovi, A., Ray, B., Lawrence, R., Takriti, S.: A logistic Regression Framework for Information Technology Outsourcing Lifecycle Management. Computer & Operations Research., Vol. 34., 3609-3627 (2007)
9. Muhammad Ali Babar, J. M., Verner, J. M., Nguyen, P. T. Establishing and maintaining trust in software outsourcing relationships: An empirical investigation. Journal of Systems and software., 80., 1438-1449 (2007)
10. Research, W.: Wolfram Mathematica. Documentation Center (2008)
11. Ross, J., Cynthia Beath, C.: Success and Failure in Outsourcing. CIO Newsletter (2006)
12. Stark, J., Arlt, M., Walker, D. H. T.: Outsourcing Decisions and Models - Some Practical Considerations for Large Organizations. In: ARLT, M. (Ed.) ICGSE '06. International Conference on Global Software Engineering (2006)
13. The-Outsourcing-Institute.: Effectively Managing The Outsourcing Relationship (2007)
14. Zhang, P., Zeng, Z.-X., Huang, C.-P.: Study on Critical Success Factors for IT Outsourcing Lifecycle. In: ZENG, Z.-X. (Ed.) International Conference on Wireless Communications, Networking and Mobile Computing, WiCom (2007)

7 Author CVs

Calvo-Manzano Jose A.

He is PhD in Computer Science. He is assistant professor in the Computer Science School at the Polytechnic University of Madrid. He is teaching in the area of Software Engineering, specifically in the domain of software process management and improvement. He has participated in more than 20 research projects (European and Spanish Public Administration). He is author of more than 50 international papers. He is author of books related to software process improvement and software engineering topics also. He is a member of SOMEPRO research group, where his main research field is process improvement, focusing in the topics of requirements management, project management, process deployment, and solicitation and supplier agreement development.

Cuevas Gonzalo

He received an Engineering degree in Telecommunications in 1965 and a PhD in Telecommunications in 1974. He also received an MS in Computer Science from the Polytechnic University of Madrid in 1972. He has been vice dean of the Computer Science faculty at the Polytechnic University of Madrid, where he is a full professor since 1970. He worked for Iberia Airlines as Software Development Director, supervising over 200 technicians, Data Processing Centre Director, Data Transmission Software Development Director, and the person in charge of strategic planning and software process improvement. He led European projects on software best practices from 1991 to 1995. His main research field is software engineering, including both technology (methods, techniques, and formalisms) and management. His current research interest is process models and methods, and transition packages. He is member of ACM, senior member of IEEE, member of the Telecommunication Engineering Association and member of the Computer Sciences Association.

Gómez Gerzon

He is a B.Sc. in Computer Science, Autonomous University of Tamaulipas (UAT), Mexico. He was assistant professor in the same university. He has occupied many management positions during more 4 years in the industry. He has published several technical papers on IT Service Management. Nowadays, he is a PhD student in the Faculty of Computer Science of the Polytechnic University of Madrid, Spain. He works in a “Metamodel for Service Level Management”.

Mejia Jezreel

He is a Master in Computer Science (2003-2005) and Computer Science Engineer (1996-2000) from the Institute Technological of Orizaba University of México. He is actually Software Engineering PhD student from the Computer Science faculty at the Polytechnic University of Madrid. His research field is Software Engineering. His current research interest consists on outsourcing process improvement. He has published several technical papers on acquisition process improvement. He is colaboring with an everis Foundation research project.

Muñoz Mirna

She is a Master in Computer Science (2002-2004) and Computer Science Engineer (1996-2000) from the Institute Technological of Orizaba University of México. She is actually Software Engineering PhD student from the Computer Science faculty at the Polytechnic University of Madrid. Her research field is Software Engineering. Her current research interest consists of the development of a methodology whose goal is to incorporate process improvement focusing on diminishing change resistance. She has published several technical papers on project management and process improvement. She is colaboring with an everis consultants research project.

San Feliu Tomás

He is PhD in Computer Science. He has been working 15 years to Software Engineering field as programmer, associate and assistant professor and consultant. He has been involved in CMM appraisals in Spain and Latin America. As Software Engineering assistant professor at the Polytechnic University of Madrid, his main research field is software process and software process improvement. His current research interest includes project management and risk management. He has published several books on process improvement and he has published technical papers in software engineering and software process improvement.

Sánchez Ángel

He was born in Madrid where he obtained his B.S. in Physics by the Complutense University and Ph.D. in Applied Physics by CSIC (Spanish National Research Council). After leaving research, Dr. Sánchez joined IBM where he was in charge of Quality and Environmental Management software-based systems. In 1998, he moved to everis, having different responsibilities in IT Governance and Quality Management. Now he is enrolled in the Innovation Department as a Development Director, in charge of identifying high potential technological opportunities for everis. He has published several articles in the field of Quantum Optics, Photonics and IT Governance.

Development of a concept for integrating various Quality Standards

Damjan Ekert[1], Philipp Hagenmeyer[2], Dr. Richard Messnarz[1]
[1]ISCN GesmbH, Austria
[2]ZF Friedrichshafen AG, Germany
dekert@iscn.com, Philipp.Hagenmeyer@zf.com, rmess@iscn.com

Abstract

This article describes how synergies and deltas between different quality models and standards have been analysed and mapped against requirements from the Automotive SPICE© and ISO/IEC 15504 process reference models. The results have been used to develop a synergetic model, to define generic requirements and questionnaires, to perform combined assessments and to develop generic templates based on best practices and requirements from different quality models.

Keywords

ISO/IEC 15504. Automotive SPICE©, ISO/TS 16494, combined assessments, synergy model, mappings, combined requirements

1 Motivation

Nowadays companies are confronted with numerous quality standards and models. The quality divisions responsible for the coordinated implementation of these quality standards and models have a more and more difficult job as the number of different models and standards as well as their complexity, are increasing. Additionally customers impose the compliance with certain standards and models so that it is no free choice and companies must implement many models at the same time.

Are this quality models and standards really very different? Actually in most cases there is a significant overlapping between the different models and standards. To take a closer look at these quality models and standards and to investigate the relationship between them (how deep they overlap or cover the same topic) a group of companies[1] from the S^{2}QI initiative formed the working group "*Modellvergleiche* – comparison of models". The major goal was defined as following:

*Show **synergies** and **deltas** between the different quality models and standards for being able to consolidate the models to a combined requirements catalogue for the processes of an enterprise*

The group of companies defined the following objectives:

- Create a mapping between the models
- Develop generic requirements and questionnaires covering a subset of quality models/ standards
- Trialling combined audits/assessments
- Define templates independent from the quality model/standard for the most common work products

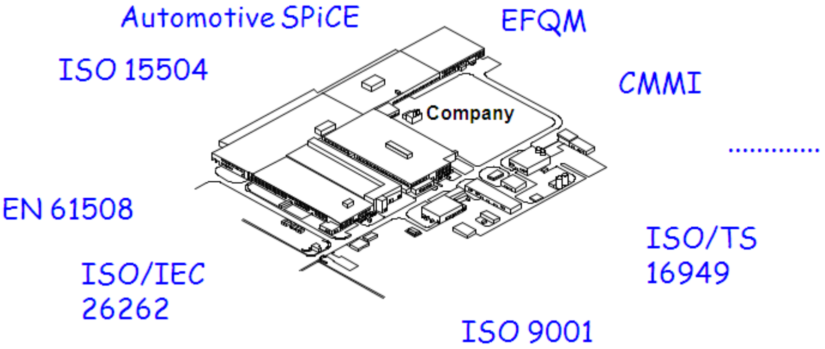


Figure 1: Current situation in companies which are confronted with many models and standards

The work started in 2007 with the foundation of the S^{2}QI initiative (System- and Software Engineering Quality Initiative). The initiative acts as a platform for companies and non-profit organizations in the

[1] MAGNA STEYR Fahrzeugtechnik AG&CoKG, Siemens AG, ISCN GesmbH, ZF Friedrichshafen AG and HM&S IT-Consulting GmbH

area of system and software engineering with the goal to increase the competitive advantage, to form win-win situations, to establish synergies between companies and to support closer collaborations in different competence networks. In the beginning the initiative covered only Austrian-Styrian companies but over the months other Austrian and German companies were invited. Currently the initiative contains the agile method development group and the model mapping group.

2 More than a simple mapping

What are the benefits of such mapping? Are there not enough mappings with CMMI vs. SPICE etc.? The objective of our initiative wasn`t just to do a simple mapping and stop there but to use the mappings as a basis for the following activities:

- Improve the internal and cross-company understanding of the quality models. In a company usually the people from the process world dealing with SPICE and CMMI are not the same people as those in the ISO 9001 world. With the mappings we could show similarities between different processes. Different quality standards, for instance, have the same requirements for a specific process but have a different level of detail. We can bring these two worlds closer together or even merge them.
- By defining company processes, specific requirements from the models and standards could be much more easily incorporated. For instance, in the first place the companies have their own requirements on how the supplier selection process is implemented. In addition to that, requirements from ISO/TS 16949 and Automotive SPICE are added to this generic company process description, and - depending on the project - the participant could already from the beginning, comply with more than one standard/model. The result would be a synergy model covering all requirements from different quality models and standards.
- By merging requirements from different quality models and standards, combined audits and assessments could be performed. In a combined assessment/audit assessors from different quality models and standards work together and perform interviews with the project participant only once. This way companies could significantly reduce the costs and time for their assessments.
- By creating mappings between different models and standards a knowledge database could be established by offering a quick overview of the quality standards and models and their requirements for a specific process. Process owners could easily check if all necessary activities were performed and all relevant work products are existing.
- New colleagues just starting the company, other companies or another company-division with previous knowledge of one quality standard or model could easily bridge the gap to the new quality model (for example coming from a CMMI based company and now dealing with Automotive SPICE© or using only ISO/TS 16949 and now confronted with Automotive SPICE©). From the mappings similarities and differences could be easily identified.
- To develop a combined questionnaire for the usage in internal and formal reviews.

Another important issue is the exchange of best practices and experiences between the members of the group in the implementation of the quality models and standards. Each participating company has its own approach and related solutions and problems. We also decided to produce templates for the most common work products with the goal that the requirements from all quality models and standards are covered.

3 Quality models

The initial step before establishing the mappings was to select reference processes (in general). Each of the participating companies wrote down their suggested processes (Figure 2). Based on a common agreement, driven by the preferred model in each company, a set of agreed processes was selected.

Automotive SPICE© based	ISO/TS16949 and EFQM based
• System requirements analysis • System architectural design • Software requirements analysis • Software design • Software construction • Software integration • Software testing • System integration • System testing • Quality assurance • Configuration management • Problem resolution management • Change request management • Project management • Supplier monitoring	• Process management • Human resource management • Training • Infrastructure • Workplace organisation • Series production • Logistic • Inspection, measuring and test equipment management • Process improvement • Knowledge management • Leadership responsibility • Social responsibility • Supplier selection • Quality management • Risk management • Requirements management • Validation • Customer support

Figure 2: Selected reference processes

After selecting the reference processes, the next step was to select the quality models and standards to be compared concerning the selected processes. Most of the companies involved in the S^{2}QI working group have experiences with Automotive SPICE©[1], a model which has been created by some of the key European automotive manufacturers (OEMs) - as an industry sector specific process reference model and process assessment model using the framework established by ISO/IEC 15504. Due to the fact that Automotive SPICE© (the mapping performed based on version 2.3) has been derived from the ISO/IEC 15504-5: 2006 standard, it is easy to provide a mapping to ISO 15504-5 at the same time [2]. The next selected model was CMMI (Capability Maturity Model® Integration, version 1.2) Continuous Representation [3]. Due to the strong automotive focus the ISO/TS16949 model was also included. ISO/TS16949:2002 is an ISO technical specification which aligns existing US, German, French and Italian automotive quality system standards within the global automotive industry. It specifies the quality system requirements for the design/development, production, installation and servicing of automotive-related products [4]. By covering ISO/TS 16949 we automatically cover the ISO 9001:2000 standard. Finally, we also took into account the EFQM Excellence Model.

In the meantime the German working group “Safety” from the “SoQrates Initiative” has also been developing a mapping between Automotive SPICE© and the Functional Safety Standard IEC 61508 with safety experts from Continental AG and ZF Friedrichshafen AG. The results will be later used in our group.

After selecting the reference processes, and the quality models/standards, we had to decide on a master reference model (to which all selected models are compared to). Because of its popularity among the involved companies Automotive SPICE© was used as the reference model. In the case that the process isn´t covered by Automotive SPICE© the ISO/TS 16494 standard is taken as the reference model.

4 Approach

Before starting with the mappings a strategy is needed to define how the models will be mapped onto each other. We decided that on the one hand we had to verify, if all requirements from the master reference model were covered, and on the other hand we also needed to investigate how deep (level of detail) these requirements were fulfilled. The following correlation symbols were used as a mapping notation:

Abbreviations:

R	Reference model
C	Compared model
n.a	Not applicable - process in the compared model does not exist

The following symbols were used to mark the coverage:
=R content from C is comparable to R
>R content from C represents more than R
<R content from C covers less than R
oR content from C isn`t covered by R
!R C misses content covered in R

The following symbols were used to mark the level of details:
+ C is more detailed then R
~ both have the same level of details
- C is less detailed then R

Topic	Reference model	A – SPICE Level 1 (HIS-Scope)	CMMI 1.2 (Cont. Represent.)	ISO/TS 16949:2002 (ISO9001:2000)	VDA 18.1 (EFQM)
System requirements analysis	A – SPICE		C = R; C - R	C = R; C ~ R	n.v.
System architectural design	A – SPICE		C = R; C ~ R	C > R; C + R	n.v.
Software requirements analysis	A – SPICE		C = R; C ~ R	C = R; C - R	n.v.
Software design	A – SPICE			C = R; C - R	n.v.
Software construction	A – SPICE			C < R; C - R	n.v.
Software integration	A – SPICE			C < R; C - R	n.v.
Software testing	A – SPICE		C = R; C - R	C < R; C - R	n.v.
System integration	A – SPICE		C = R; C - R	C < R; C - R	n.v.
System testing	A – SPICE		C = R; C - R	C = R; C - R	n.v.
Quality assurance	A – SPICE			C < R; C - R	n.v.
Configuration Management	A – SPICE			C < R; C - R	n.v.
Problem resolution management	A – SPICE			C < R; C - R	n.v.
Change request management	A – SPICE			C < R; C - R	n.v.
Project management	A – SPICE		C > R; C ~ R	C < R; C - R	C > R; C - R
Supplier Monitoring	A – SPICE			C = R; C ~ R	C > R; C - R

Figure 3: Extract from the mapping

The goal for the first phase was to establish all mappings with Automotive SPICE© as the reference model (to which other models are compared to). In the second phase we have planned to continue with the other (for the moment non-selected) processes and afterwards add new models.

5 Mappings

For each of the processes (Figure 3) a more detailed mapping has been established. The mappings are based upon the Automotive SPICE© base practices. For each base practice the corresponding specific practice (in case of CMMI) or chapter/subchapter (in case of ISO/TS 16949 and EFQM) has been listed. One base practice can be also covered by more than one item. In the following examples mappings between Automotive SPICE© and ISO/TS 16949 as well as Automotive SPICE© and CMMI are illustrated:

Coverage	Level of details
Rating : [<R]	Rating: [-]
Explanation: Activities concerning planning, scheduling and reporting back to the initiator of the change request is not covered in detail by the ISO/TS 16949	Explanation: The level of detail concerning dependencies and relationships to other change requests, allocation of resources, scheduling and reporting is requested in Automotive SPICE© in a much higher detail.
BP1 => 7.1.4 (=R)	BP1 => 7.1.4 (~)
BP2 => 7.1.4 (=R)	BP2 => 7.1.4 (~)
BP3 => 7.3.7 (=R)	BP3 => 7.3.7 (~)
BP4 => 7.3.7 (=R)	BP4 => 7.3.7 (~)
BP5 => 7.1.4 (=R)	BP5 => 7.1.4 (-)
BP6 => 7.1.4 und 7.3.7(=R)	BP6 => 7.1.4 und 7.3.7 (~)
BP7 => 7.3.7 (<R)	BP7 => 7.3.7 (-)
BP8 => 7.3.7 (=R)	BP8 => 7.3.7 (-)
BP9 => 7.1.4 (=R)	BP9 => 7.1.4 (~)
BP10 => (<R)	BP10 => (-)
BP11 => 7.3.7 (=R)	BP11 => 7.3.7 (~)
BP12 => (!R)	BP12 => (-)

Table 1: Change Request Management - Automotive SPICE© vs. ISO/TS 16949

The mapping in Table 1 illustrates that Automotive SPICE© covers a wider field (more areas) with a higher level of detail for the process Change Request Management. For a company already fulfilling the requirements from the ISO 9001:2000 or ISO/TS16949 and now implementing Automotive SPICE©, the mapping shows the gap between the two standards and which base practices aren`t covered by the ISO 9001:2000 and ISO/TS16949. With the help of the mappings the transition could be much smoother.

If we apply this comparison on other development processes such as Software Testing, the mapping would show some missing requirements in the ISO 9001:2000 or ISO/TS16949. One missing practice, for example, would be the bidirectional traceability. Also, in general, Automotive SPICE© has a strong focus on software where ISO 9001:2000 or ISO/TS16949 are more general and focus towards a production process. In most cases the ISO 9001:2000 or ISO/TS16949 lacks detail when it comes to software development related engineering processes.

Another mapping we considered were the mappings from Automotive SPICE© to CMMI. Some processes like Project Management are covered in CMMI by two processes (Project Planning and Project Monitoring and Control). In the case of project management, the CMMI content covers are wider scope.

6 Implementation Experiences

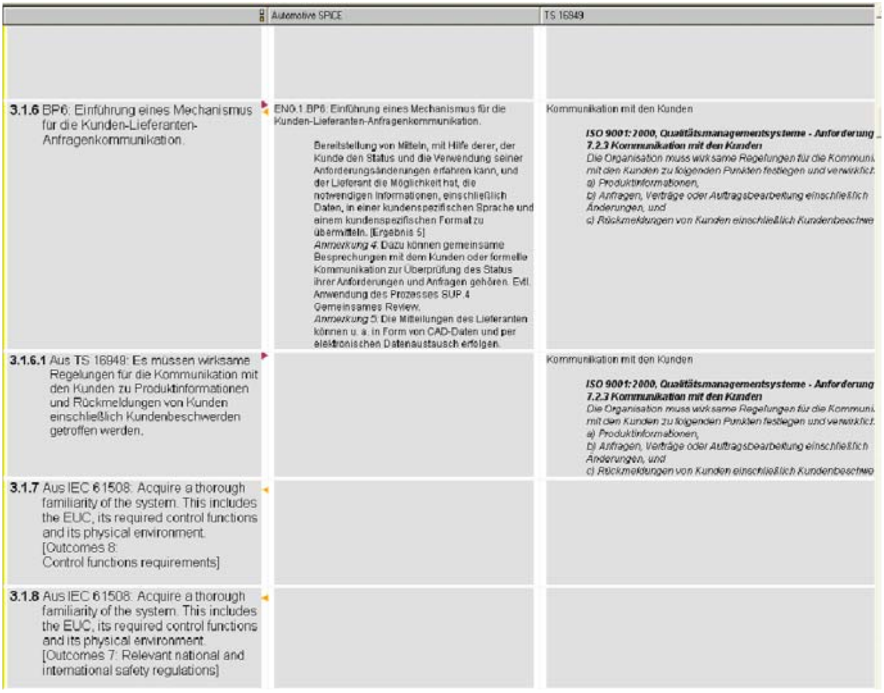
The following chapter describes how the intermediate results from the initiative were used by the partnering companies. The mapping results illustrated in Figure 4 are used in an internal knowledge database, extending the process-requirements from Automotive SPICE© with matching requirements from other quality models.

Thus the organization is aware of the consolidated requirements to its processes and can ensure that all these process-requirements are realized in process instructions without gaps or redundancies.

Another implementation case is illustrated in Figure 5 and Figure 6 where combined assessments are already preconfigured in an assessment portal system the Capability Adviser Assessment Tool. The assessment is based on Automotive SPICE© and additional requirements from the other standards are displayed on request. This approach can be used for self-assessment, where the tool acts like a knowledge database as well as in formal assessments. Good results have been already achieved by combining ISO15504 and ISO 9001:2000 (Figure 6). The assessment approach and result has been also approved by an ISO 9001 certification body.

The Functional Safety working group in the German SoQrates[2] Initiative is currently also finalizing its mapping between Automotive SPICE© and IEC 61508. Figure 5 shows such a combined assessment approach.

Similar combined assessments could be performed in future with Automotive SPICE©/CMMI and ISO/TS 16949.



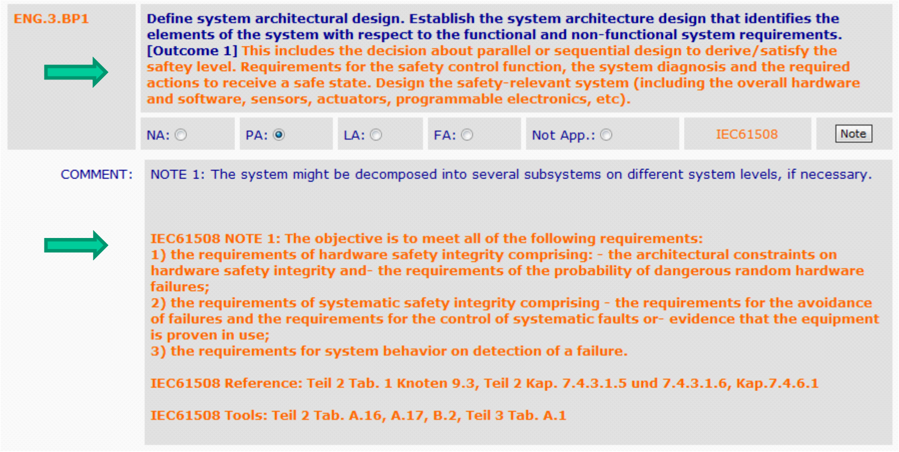
	Automotive SPICE	TS 16949
3.1.6 BP6: Einführung eines Mechanismus für die Kunden-Lieferanten-Anfragenkommunikation.	ENG.1 BP6: Einführung eines Mechanismus für die Kunden-Lieferanten-Anfragenkommunikation. Bereitstellung von Mitteln, mit Hilfe derer, der Kunde den Status und die Verwendung seiner Anforderungsänderungen erfahren kann, und der Lieferant die Möglichkeit hat, die notwendigen Informationen, einschließlich Daten, in einer kundenspezifischen Sprache und einem kundenspezifischen Format zu übermitteln. [Ergebnis 5] *Anmerkung 4*: Dazu können gemeinsame Besprechungen mit dem Kunden oder formelle Kommunikation zur Überprüfung des Status ihrer Anforderungen und Anfragen gehören. Evtl. Anwendung des Prozesses SUP.4 Gemeinsames Review. *Anmerkung 5*: Die Mitteilungen des Lieferanten können u. a. in Form von CAD-Daten und per elektronischen Datenaustausch erfolgen.	Kommunikation mit den Kunden ***ISO 9001:2000, Qualitätsmanagementsysteme - Anforderung*** ***7.2.3 Kommunikation mit den Kunden*** *Die Organisation muss wirksame Regelungen für die Kommuni* *mit den Kunden zu folgenden Punkten festlegen und verwirklict* *a) Produktinformationen,* *b) Anfragen, Verträge oder Auftragsbearbeitung einschließlich Änderungen, und* *c) Rückmeldungen von Kunden einschließlich Kundenbeschwe*
3.1.6.1 Aus TS 16949: Es müssen wirksame Regelungen für die Kommunikation mit den Kunden zu Produktinformationen und Rückmeldungen von Kunden einschließlich Kundenbeschwerden getroffen werden.		Kommunikation mit den Kunden ***ISO 9001:2000, Qualitätsmanagementsysteme - Anforderung*** ***7.2.3 Kommunikation mit den Kunden*** *Die Organisation muss wirksame Regelungen für die Kommuni* *mit den Kunden zu folgenden Punkten festlegen und verwirklict* *a) Produktinformationen,* *b) Anfragen, Verträge oder Auftragsbearbeitung einschließlich Änderungen, und* *c) Rückmeldungen von Kunden einschließlich Kundenbeschwe*
3.1.7 Aus IEC 61508: Acquire a thorough familiarity of the system. This includes the EUC, its required control functions and its physical environment. [Outcomes 8: Control functions requirements]		
3.1.8 Aus IEC 61508: Acquire a thorough familiarity of the system. This includes the EUC, its required control functions and its physical environment. [Outcomes 7: Relevant national and international safety regulations]		

Figure 4: Knowledge database with the requirements from different quality standards

Extract from Figure 4:

3.1.6 Automotive SPICE© ENG.1. BP6 Establish customer-supplier query communication mechanism	*Customer Communication* *ISO9001:2000: 7.2.2 Review of Requirements Related to the Product*
3.1.6.1 From the ISO/TS16949 Communication arrangements with customers must be established to ensure understanding of product and contract requirements and facilitate feedback, including complaints.	*Customer Communication* *ISO9001:2000: 7.2.3 Customer communication*

[2] Soqrates (Software Quality Rates Maturity), http://www.isqi.org/en/soqrates/



ENG.3.BP1

Define system architectural design. Establish the system architecture design that identifies the elements of the system with respect to the functional and non-functional system requirements. [Outcome 1] This includes the decision about parallel or sequential design to derive/satisfy the saftey level. Requirements for the safety control function, the system diagnosis and the required actions to receive a safe state. Design the safety-relevant system (including the overall hardware and software, sensors, actuators, programmable electronics, etc).

NA: ○ | PA: ● | LA: ○ | FA: ○ | Not App.: ○ | IEC61508 | Note

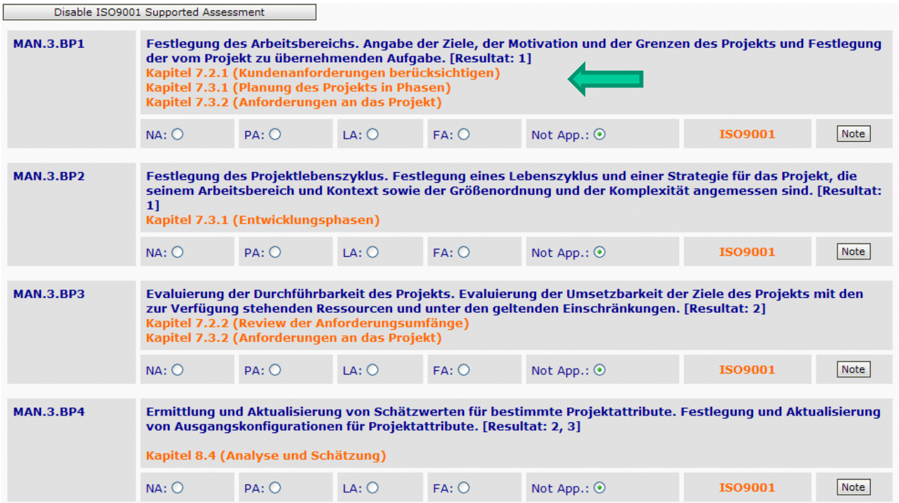
COMMENT: NOTE 1: The system might be decomposed into several subsystems on different system levels, if necessary.

IEC61508 NOTE 1: The objective is to meet all of the following requirements:
1) the requirements of hardware safety integrity comprising: - the architectural constraints on hardware safety integrity and- the requirements of the probability of dangerous random hardware failures;
2) the requirements of systematic safety integrity comprising - the requirements for the avoidance of failures and the requirements for the control of systematic faults or- evidence that the equipment is proven in use;
3) the requirements for system behavior on detection of a failure.

IEC61508 Reference: Teil 2 Tab. 1 Knoten 9.3, Teil 2 Kap. 7.4.3.1.5 und 7.4.3.1.6, Kap.7.4.6.1

IEC61508 Tools: Teil 2 Tab. A.16, A.17, B.2, Teil 3 Tab. A.1

Figure 5: Combined Assessment Automotive SPICE and Safety (IEC61508)



Disable ISO9001 Supported Assessment

MAN.3.BP1

Festlegung des Arbeitsbereichs. Angabe der Ziele, der Motivation und der Grenzen des Projekts und Festlegung der vom Projekt zu übernehmenden Aufgabe. [Resultat: 1]
Kapitel 7.2.1 (Kundenanforderungen berücksichtigen)
Kapitel 7.3.1 (Planung des Projekts in Phasen)
Kapitel 7.3.2 (Anforderungen an das Projekt)

NA: ○ | PA: ○ | LA: ○ | FA: ○ | Not App.: ● | ISO9001 | Note

MAN.3.BP2

Festlegung des Projektlebenszyklus. Festlegung eines Lebenszyklus und einer Strategie für das Projekt, die seinem Arbeitsbereich und Kontext sowie der Größenordnung und der Komplexität angemessen sind. [Resultat: 1]
Kapitel 7.3.1 (Entwicklungsphasen)

NA: ○ | PA: ○ | LA: ○ | FA: ○ | Not App.: ● | ISO9001 | Note

MAN.3.BP3

Evaluierung der Durchführbarkeit des Projekts. Evaluierung der Umsetzbarkeit der Ziele des Projekts mit den zur Verfügung stehenden Ressourcen und unter den geltenden Einschränkungen. [Resultat: 2]
Kapitel 7.2.2 (Review der Anforderungsumfänge)
Kapitel 7.3.2 (Anforderungen an das Projekt)

NA: ○ | PA: ○ | LA: ○ | FA: ○ | Not App.: ● | ISO9001 | Note

MAN.3.BP4

Ermittlung und Aktualisierung von Schätzwerten für bestimmte Projektattribute. Festlegung und Aktualisierung von Ausgangskonfigurationen für Projektattribute. [Resultat: 2, 3]

Kapitel 8.4 (Analyse und Schätzung)

NA: ○ | PA: ○ | LA: ○ | FA: ○ | Not App.: ● | ISO9001 | Note

Figure 6: Combined Assessment ISO 9001:2000 and ISO/IEC 15504

7 Conclusion and Outlook

As stated above, the results of the initiative are already applied in different tools in the partnering companies. The group is currently finishing the first phase of the mappings (with Automotive SPICE© as the reference model).

Our next objective is to review the existing mappings from other companies, groups and experts who are not involved in the initiative. Based on the reviewed mappings, combined questionnaires for internal reviews will be developed. The questionnaires will help the quality department in their review process as well as the project members and process owners to prepare themselves.

The initiative also invites new partners to join the group and act either as reviewers or developers of new mappings based on the existing or new quality models and standards.

8 Literature

[1] Automotive SPICE©, Version 2.3, September 2007, ISSN 0943-9412, http://www.automotivespice.com/

[2] ISO/IEC 15504-5:2006 Information technology -- Process Assessment -- Part 5: An exemplar Process Assessment Model. International Organization for Standardization (2006)

[3] Software Engineering Institute, CMMI for Development, Version 1.2, CMU/SEI 2006-TR-008 ESC-TR-2006-008

[4] ISO/TS 16949:2002 Technical specification, second edition 2002-03-01

[5] EFQM Excellence Model, VDA18.1, ISSN 0943-9412

[6] R. Messnarz, et. al, Integrated Automotive SPICE and Safety Assessments, in: Proceedings of the EuroSPI 2008 Conference, Dublin, Ireland, Oct 2008, ISBN: 978-87-7398-150-4

9 Author CVs

Dipl.Ing. Damjan Ekert

Dipl.-Ing. Damjan Ekert studied Telematics at the University of Technology, Graz, Austria. He finished his studies with distinction. Since 2001 he works for ISCN as SW project manager and SW integrator. He is a certified e-Security Manager and European Project Manager Trainer. For the last three year he has been working as a consultant in the field of Software Process Improvement, he is member of the S2QI Initiative, certified ISO15504 Assessor, member of the European Certification and Qualification Association and vice- president of the ISECMA e.V. (International IT Security Management Board).

Dipl.Ing. Philipp Hagenmeyer

Philipp Hagenmeyer has studied Mechanical Engineering focusing on driveline- and combustion technology. He graduated from the University of Karlsruhe, Germany, in 2006. Various internships in the Automotive Industry (process optimization projects) made him start his career at the ZF Friedrichshafen AG in 2007 as a process-optimization-engineer for system-, function- and software development processes. His main fields of application are Requirement- and Configuration Management. Philipp is an iNTACS™ certified ISO/IEC 15504 Provisional Assessor for AutomotiveSPICE® and a member of the S2QI Initiative.

Dr. Richard Messnarz

Dr. Richard Messnarz (rmess@iscn.com) is the Executive Director of ISCN LTD. He studied at the University of Technology Graz and he worked as a researcher and lecturer at this University from 1991 - 1996. In 2 European mobility projects (1993 and 1994) he was involved in the foundation of ISCN, and he became the director of ISCN in 1997.

He is the editor of a book "Better Software Practice for Business Benefit", which has been published by IEEE (www.ieee.org) in 1999 (the leading research publisher in the USA). He is the chairman of the EuroSPI initiative and chair of the programme committee of the EuroSPI conference series.

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

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

A Method Framework for Engineering Process Capability Models

Clenio F. Salviano[1], Alessandra Zoucas[2], Jorge V. L. Silva[1], Angela M. Alves[1],
Christiane G. von Wangenheim[2,3], and Marcello Thiry[2]

[1] CTI: Centro de Tecnologia da Informação Renato Archer
Rodovia D. Pedro I, km 143.6, CEP 13069-901, Campinas, SP, Brazil
clenio.salviano@cti.gov.br, clenio.salviano@gmail.com, jorge.silva@cti.gov.br,
angela.alves@cti.gov.br

[2] Universidade do Vale do Itajaí (UNIVALI) – Campus São José
Mestrado em Computação Aplicada, Rod. SC 407, Km 04, 88122-000 São José, SC, Brazil;
alessandrazoucas@gmail.com, marcello.thiry@gmail.com, gresse@gmail.com

[3] Universidade Federal de Santa Catarina (UFSC)
Programa de Pós-Graduação em Ciência da Computação, Florianópolis, SC, Brazil
gresse@gmail.com

Abstract

Software Process Improvement, based on a Maturity Level or a Process Capability Profile, from a capability maturity model or an ISO/IEC 15504-based model, is well established in the software industry as a successful mean for improving software intensive organizations. In consequence there is an opportunity to understand how these models have been developed and consolidate this knowledge to support the development of new models by, among others, the industry. This article introduces a Method Framework for Engineering Process Capability Models as an element of a methodology on a Process Capability Profile to drive Process Improvement. This method framework is based on five previous successful experiences in which we experiment different processes to develop different process capability models. The current version is composed of sequential practices, customization rules, examples of utilization and examples of techniques. An initial validation indicates a first confidence that this method framework is a useful proposal for developing methods and processes for engineering process capability models.

Keywords

Software Process Improvement (SPI), Process Capability Model, PRO2PI Methodology, CMMI, ISO/IEC 15504

1 Introduction

Around the 1980´s, Watts Humphrey and others at the Software Engineering Institute (SEI) elicited and generalized good practices from few software intensive organizations that had been working well. Those practices were organized as sequential and cumulative maturity level as the Capability Maturity Model for Software (CMM or SW-CMM) [1]. With the success of CMM as practical guidelines for a feasible practical improvement of software intensive organizations, a new area emerged: Software Process Improvement (SPI). As an evolution of CMM two frameworks of models were established: ISO/IEC 15504 International Standard for Process Assessment [6] and the Capability Maturity Model Integration (CMMI) [2]. CMMI is aligned with ISO/IEC 15504 and the CMMI-DEV model [2] is the successor of CMM.

Basically the current SPI area continues the same as it was established around CMM and the CMMI-DEV is the dominant model, although ISO/IEC 15504-based models are relevant too. There are, however, forces around the successful current SPI that urge for a revision and evolution of SPI area [3, 4]. One of these forces is related with the need to develop more process capability models. Therefore there is an opportunity to understand how process capability models have been developed and consolidate this knowledge to support the development of new models. The industry will participate more in this development, as models for more specific business context, for more specific domain or even for a specific organization will be develop as customization of relevant more generic models.

The term **Process Capability Model** [5] is used to mean models of best practices organized with the concepts of process capability and process maturity. In this sense a capability maturity model, as, for example, the CMMI-DEV model [2], is a process capability model. An ISO/IEC Process Assessment Model, as, for example, the ISO/IEC 15504-5 model [6], is a process capability model as well.

In order to support the development of new models, this article introduces PRO2PI-MFMOD as a "**M**ethod **F**ramework for Engineering Process Capability **Mod**els as an element of the **PRO2PI** Methodology". PRO2PI is a methodology on "***Pro***cess Capability ***Pro***file ***to*** drive **P**rocess **I**mprovement" [3, 4]. PRO2PI is an exemplar methodology for a proposed evolution of current SPI area, named MDPEK: "(Process Capability Profile) **M**odel-**D**riven (**P**rocess Capability **E**ngineering) for (Software, System and other **K**nowledge Working) Intensive Organization" [3].

The initial objective was to develop a method. During the construction, we realize that the variety of situations, however, raised significant risks to develop a single method. Therefore, we decided to develop a more abstract methodological element to support the definition of methods. We developed a **Method Framework**. This term is already used with similar meaning, similar objective and similar reasons for the Method Framework for Engineering System Architectures (MFESA) [16] which confirms the usage of this term within this context. The major difference from the meaning of method framework is that PRO2PI-MFMOD does not define the contextual elements because these elements are already provided by the PRO2PI methodology.

This article is organized as follows. This first section provides an introduction to the article. The second section digests the PRO2PI Methodology. The third section establishes goals, methodology and process for the development of this method framework. The fourth section reviews previous experiences in developing models. The fifth section introduces the PRO2PI-MFMOD. The sixth section presents how those processes can be considered as examples of the method framework. The seventh section describes how the method framework is planning to be used for a complex system. Finally, the eighth section presents some initial validation and some conclusions.

2 PRO2PI Methodology

PRO2PI is a multi-model process improvement methodology driven by process capability profiles. As an exemplar methodology for MDPEK, PRO2PI supports process improvement using elements from multiples reference models and other sources. These elements are selected or defined and they are integrated as process capability profile. A process capability profile that drives a process improvement under PRO2PI methodology is also named as a PRO2PI. Figure 1 presents the conceptual elements of the PRO2PI methodology, the relationship among them and the name of each one.

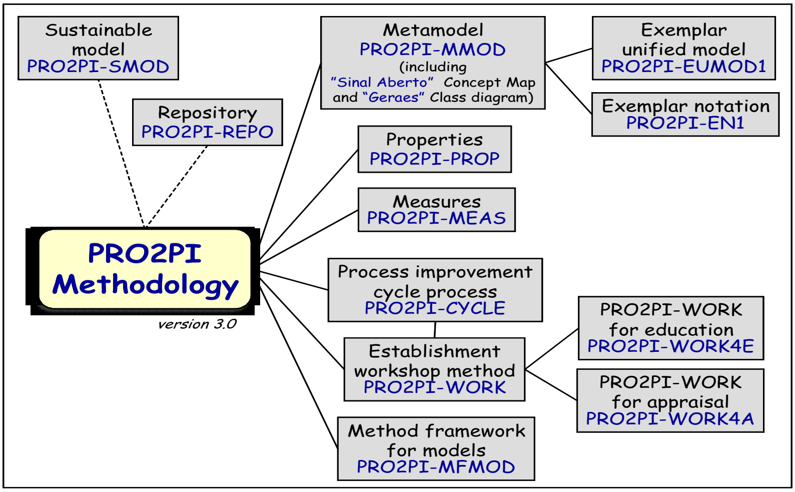

Sustainable model PRO2PI-SMOD
Repository PRO2PI-REPO
Metamodel PRO2PI-MMOD (including "Sinal Aberto" Concept Map and "Geraes" Class diagram)
Exemplar unified model PRO2PI-EUMOD1
Exemplar notation PRO2PI-EN1
Properties PRO2PI-PROP
Measures PRO2PI-MEAS
PRO2PI Methodology
version 3.0
Process improvement cycle process PRO2PI-CYCLE
Establishment workshop method PRO2PI-WORK
PRO2PI-WORK for education PRO2PI-WORK4E
PRO2PI-WORK for appraisal PRO2PI-WORK4A
Method framework for models PRO2PI-MFMOD

Figure 1 – PRO2PI methodology elements

PRO2PI-SMOD is a sustainable model for the dissemination and evolution of PRO2PI methodology. PRO2PI-REPO is a repository for PRO2PI assets. PRO2PI-MMOD is a metamodel for a process capability profile and process capability model. Using PRO2PI-MMOD, PRO2PI-EUMOD1 is an exemplar unified process capability model with elements from selected relevant models, and PRO2PI-EN1 is a notation to represent a PRO2PI. PRO2PI-PROP is a set of properties for a PRO2PI. PRO2PI-MEAS is a set of measures to qualify a PRO2PI. PRO2PI-CYCLE is a process for process improvement cycles including a function to define, update or use a PRO2PI.

PRO2PI-WORK is a method for a workshop to establish a process capability profile to drive a process improvement cycle. This method was developed to guide the implementation of the first three phases of PRO2PI-CYCLE in a low capability, small organization. In addition, two customized variations of this method were defined. PRO2PI-WORK4A is a method for a workshop with emphasis in the assessment of current practices and PRO2PI-WORK4E is a method for a workshop with emphasis in education on process improvement. PRO2PI-MFMOD is a method framework for engineering process capability models that is described in this article.

3 Goals, Methodology and Process

This section establishes a main general goal, three derived objective goals, the methodology and the process used to guide the development of the method framework. The main general goal is that the method framework is a useful proposal for developing methods and processes for engineering Process capability models. The first objective goal (Goal G1) is that the method framework could be considered as a generalization of a given set of processes and methods used to successfully develop process capability models. The second objective goal (Goal G2) is that it is part of the PRO2PI methodology [3, 4] because developing models is part of the scope of this methodology. The third unfolded goal (Goal G3) is that it supports the planning for a process to develop a model for best practices in a given complex system.

The development of this method framework followed the process capability levels form ISO/IEC 15504 as a methodology [6]. First we participated and studied successful processes to develop models in order to construct knowledge about developing models. This is related with capability level 1 for a "process capability model engineering" process area. Then we planned, performed, monitored and controlled five successful processes to develop five different process capability models. This is related with capability level 2 for this same process area. The development of this method framework from an analysis of these five previous successful experiences in model development prepare for capability level 3. The engineering of a process capability model will be guided by a planned, performed, monitored and controlled defined process that is tailored from the method framework. Therefore, the defined process will be a capability level 3 process.

Using this methodology, a process was planned and performed with the following seven activities to develop the method framework presented in this article: (1) preparation for the work; (2) identification and initial analyses of previous experience from our research group and from others groups; (3) revision of PRO2PI methodology to include the method framework; (4) development of a preliminary version of the method framework; (5) more disciplined revision of the previous experiences identifying including a relationship between the process used in each previous experience with the preliminary method framework; (6) revision of the method framework in such way that all previous experiences could be considered as examples of instantiation of this method framework; and (7) usage of the method framework to plan a process to develop a process to develop a process capability model for a complex system.

4 Structured review of previous experiences

This section reviews five previous successful experiences in which we experiment different processes to develop different process capability models. In addition four more experiences from others are presented. We also participated in some of these experiences from others. For each one of these nine experiences a structured review is presented with a phrase name (**in bold type**), a brief description, the activities of the actual planned and performed process used to develop the model, and examples of techniques used to develop the model.

Process for a model for education: This model was composed of a new process area to cover the teaching of a technical course [7]. This process area is defined as a new process for the ISO/IEC 15504-5 model. The strategy was to abstract a process area from the current process used by the teacher. For the development of this process capability model for education, a process with the following seen activities was defined and used: (1) description of the current process used by the teacher; (2) analyses of the guidelines defined by the organization; (3) description of an improved process, following the ISO/IEC 15504-5 model, to be used by the teacher; (4) definition of a new process area for ISO/IEC 15504-5 such that improved process is an exemplar implementation; (5) assessment of the current process; and (6) revision and consolidation of the new process area. A specific technique predefined for this process is to abstract a process area from an actual process.

Process for the MARES model: A specialization of the ISO/IEC 15504-5 model for Small and Medium Enterprises (SME) was developed as part of o project to develop a Method for Process Assessment in Small Software Companies (MARES) [8]. A process for the MARES Model, with seven activities, was planned and followed: (1) state of the art of process improvement in SME review and study of ISO/IEC 15504-5; (2) state of the art of methods and models for SPI in SME; (3) requirements definition for the proposed model; (4) development of a draft model; (5) evaluation through four case studies using the draft model; (6) revised draft model; and (7) evaluation through two new case studies. Two specific techniques predefined for this process are state of the art literature review to gain knowledge and case studies to validate a draft model.

Process for a CMMI specialization to CBSE: For a development of a process capability model for Component Based Software Engineering (CBSE) a process was defined and used [9]. The eight activities of this process are as follows: (1) review the state of the art and state of the practice, in this case, for CBSE, (2) identify a process capability model more appropriate to be specialized for the domain (in this case CBSE), (3) identify or define a set of additional process areas to cover the major CBSE specific aspects, (4) represent these new process areas using the format of the base model, (5) identify process areas from the base model that needs customizations for CBSE and perform those customizations (6) identify other generic process areas from other relevant models that are relevant for the domain and include them in the model, (7) consider practices from relevant organization that already implement good CBSE, include those practices as additional sources, and revise the model to cover these practices, and (8) use the model in CBSE organizations, analyse the results and revise the model. A specific technique predefined for this process is to translate process areas from a given model (in this case the ISO/IEC 15504-5 model) to new process areas for another model (in this case the CMMI-DEV model).

Process for a CMMI specialization to banking domain: In the development of a specialization of the CMMI-DEV process capability model for software development in the banking domain [10], a process for a CMMI model specialization was defined and used with the following seven activities: (1) characterization of the domain, (2) selection of some process areas, (3) initial description of the domain, (4) exploration of the domain description and specialization of the selected process areas, (5) revision of the domain description and the process areas specialization, (6) validation; and (7) revision and consolidation. A specific technique predefined for this process is to describe a domain using phrases and to relate them to some practices of a model in order to determine if a practice from a model has higher, same or less relevance for that domain.

Process for the SPICE for Research model: For developing an ISO/IEC 15504-based process capability model for University Research Laboratory (SPICE for Research Model) [11, 12] a process was defined and used for the construction of this model. The six activities of this process are as follows: (1) state of the art review, (2) best practices survey, (3) process capability model draft design, (4) process capability model draft development, (5) process capability model validation, and (6) process capability model version 1.0. University Research Laboratory (URLab) is a unique environment that performs knowledge-intensive activities. The SPICE for Research considers the best practices investigated in some URLabs and the technical and scientific literature on knowledge management, research management, organizational management, and capability models. Two different communities validated SPICE for Research: the community of managers of research and the community of researchers with experience in process improvement [12]. Two specific techniques predefined for this process are using questionnaires to obtain information from experts in the domain and performing extensive literature review to understand best practices for the domain.

Generic process for consolidated models: There are a set of process capability models that can be considered as more relevant and more consolidated models, including the original SW-CMM model, CMMI models (CMMI-DEV, CMMI-SRV and CMMI-ACQ), ISO/IEC 15504 models (ISO/IEC 15504-5 and ISO/IEC 15504-6), other ISO/IEC conformant models (OOSPICE, Automotive SPICE, Enterprise SPICE and others), the e-SCM models, the MPS.BR model and the COMPETISOFT model. For neither one of them, we could found a complete documented process about how each one was developed. There are only general information about the development, as, for example, the ISO rules and procedures to develop an International Standard. Up to now, we did not produce activities for the process used to develop these models.

Process for a leadership model: In a development of a process capability model for leadership of Integrated Virtual Teams, Tuffley [13] defined and used a process with the following five activities: (1) literature review; (2) process capability model draft development; (3) cases study using the draft model (4) results analyses and (5) model consolidation (with possible cycles of activities 2, 3 and 4).

Process for models from requirements transformation: Barafort et alli proposed a method to transform a set of requirements into a process capability model [14]. They followed this method to develop a process capability model for IT Service Management from the ISO 20000 requirements. This method has the following nine activities: “(1) identify elementary requirements in a collection of requirements, (2) organize and structure the requirements, (3) identify common purposes upon those requirements and organize them towards domain goals, (4) identify and factorize outcomes from the common purposes and attach them to the related goals, (5) group activities together under a practice and attach it to the related outcomes, (6) allocate each practice to a specific capability level, (7) phrase outcomes and process purpose, (8) phrase the base practices attached to outcomes, and (9) determine work products among the inputs and outputs of the practices” [14].

Process for a model for SaaS: Cancian developed a draft process capability model as a reference guide for assessing software development process practiced by SaaS (Software as a Service) providers [15]. In order to accomplish its objectives, quality requirements that providers should meet were elicited. After having been summarized and analyzed, the requirements were mapped to existing standards and reference models. From this mapping, a reference guide was proposed. A process was defined and used for the construction of this draft model, with the following five activities: (1) literature review, (2) gathering of requirements, (3) complementation and determination of the priority among those requirements, (4) mapping of those requirements, and (5) construction of the reference guide.

5 PRO2PI-MFMOD Method Framework

PRO2PI-MFMOD is a method framework for engineering process capability models based on context and characteristics of a segment or domain. The current version is composed of four types of elements, each one, by a coincidence, with seven elements: sequential practices, customization rules, examples of utilization and examples of techniques. The examples of utilization and examples of techniques are described in Section 4. The sequential practices and customization rules are described in this section.

PRO2PI-MFMOD defines seven sequential practices to guide the development of a method or a process to develop a process capability model: (1) initial decisions, (2) sources analysis, (3) strategy for development, (4) model design, (5) draft model development, (6) draft model validation, and (7) model consolidation (Figure 2).

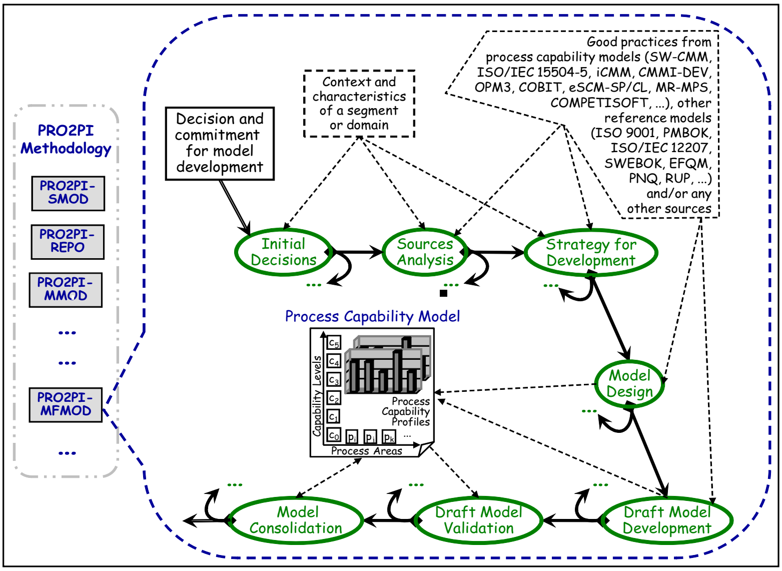

PRO2PI Methodology
PRO2PI-SMOD
PRO2PI-REPO
PRO2PI-MMOD
...
...
PRO2PI-MFMOD
...
Decision and commitment for model development
Context and characteristics of a segment or domain
Good practices from process capability models (SW-CMM, ISO/IEC 15504-5, iCMM, CMMI-DEV, OPM3, COBIT, eSCM-SP/CL, MR-MPS, COMPETISOFT, ...), other reference models (ISO 9001, PMBOK, ISO/IEC 12207, SWEBOK, EFQM, PNQ, RUP, ...) and/or any other sources
Initial Decisions
Sources Analysis
Strategy for Development
...
Process Capability Model
Capability Levels
c5
c4
c3
c2
c1
c0
pi
pj
pk
...
Process Capability Profiles
Process Areas
Model Design
Draft Model Development
Draft Model Validation
Model Consolidation

Figure 2- PRO2PI-MFMOD´s seven sequential practices

The first practice of PRO2PI-MFMOD is related with some initial decisions after a decision and commitment for model development. These initial decisions can be related with any one of the following six practices. In the second practice (Sources analysis) we identified, gather and analysed sources for good practices. These sources can include literature review, surveys, and others. These sources are based on the context and characteristics of a segment or domain. The third practice (Strategy for development) is related with the definition of the strategy to be used to develop the model. One key issue is how the community of interest will be involved in this development. Another issue is using selected good practices from process capability models (SW-CMM, ISO/IEC 15504-5, iCMM, CMMI-DEV, OPM3, COBIT, eSCM-SP/CL, MR-MPS, COMPETISOFT, ...), other reference models (ISO 9001, PMBOK, ISO/IEC 12207, SWEBOK, EFQM, PNQ, RUP, ...) and/or any other sources.

The fourth practice (Model design) is related with the design of the process capability model. ISO/IEC 15504 establish as general structure for model design as Process Reference Model and Process Assessment Model. PRO2PI-MMOD as a metamodel provides a reference for this design. The fifth practice is the draft model development. The sixth practice is the validation of the draft model. The seventh practice is the consolidation of the process capability model.

As part of the method framework, these seven sequential practices must be customized as activities of a method or even by a process. This customization is oriented by combinations of seven simple customization rules (CR1 to CR7). These seven customization rules are described as follows, in terms of the relationship between one or more method framework´s practice and one or more method or process´s activity:

CR1: A practice corresponds to an activity (*one practice to one activity*);

CR2: There is no activity that corresponds to a practice, because the results to be produced by the practice execution are already predefined by the method or process (*one practice to zero activity*);

CR3: There are no activities that correspond to one or more consecutives final practices, because the life cycle of the method or process ends before those final practices (*many final practices to zero activity*);

CR4: Two or more activities correspond to one practice, because the activities are more detailed customization of the practice (*one practice to many activities*);

CR5: An activity corresponds to two or more consecutive practices, because the activity is a more general and simplified customization of the practices (*many practices to one activity*);

CR6: There are consecutive activities that correspond to cycles of consecutive practices (*many practices to activity cycles*); and

CR7: There is one or more technique that is specified for one or more activities.

The next provides representations of those processes (described in Section 4) as customizations of the method framework and explain these customizations in terms of applications of these customizations rules. In this way, the next section supports the understanding of these customizations rules.

6 Processes and PRO2PI-MFMOD

Table 1 show the PRO2PI-MFMOD´s seven practices and the activities of each one of the five processes described in Section 4 and indicate how each practice is related with the activities.

Table 1 – practices of PRO2PI-MFMOD and activities of five processes

Process for	Prac. 1	Prac. 2	Prac. 3	Prac. 4	Prac. 5	Prac. 6	Prac. 7
PRO2PI-MFMOD	1	2	3	4	5	6	7
a model for education		1 2 3	4			5 6	
the MARES model		1 2	3	4		5	6 7
a CMMI specialization to CBSE		1 7	2	3 5 6	4	8	
a CMMI specialization to banking		1		2 3	4 5	6	7
the *SPICE for Research* model		1 2		3	4	5	6

The process for a model for education customizes the method framework applying the following customization rules: (a) rule CR2 is applied related with practice 1 because the initial decisions were already taken before the process was defined; (b) rule CR4 is applied because the activities 1, 2 and 3 are more detailed than the correspondent practice 2; (c) rule CR5 is applied because the activity 4 is more general and simple than the correspondents practices 3, 4 and 5; (d) rule CR4 is applied again because the activities 5 and 6 are more detailed than the correspondent practice 6; and (e) rule CR7 is applied because the process finished with the validation of the model draft version, and then there is no activity that correspond to the final practice 7.

The process for the MARES mode customizes the method framework applying the following customization rules: (a) rule CR2 is applied related with practice 1 because the initial decisions were already taken before the process was defined; (b) rule CR4 is applied because the activities 1 and 2 are more detailed than the correspondent practice 2; (c) rule CR1 is applied because the activity 3 corresponds to practice 3; (d) rule CR5 is applied because the activity 4 is more general and simple than the correspondents practices 4 and 5; (e) rule CR1 is applied because the activity 5 corresponds to practice 6; and (f) rule CR4 is applied again because the activities 6 and 7 are more detailed than the correspondent practice 7.

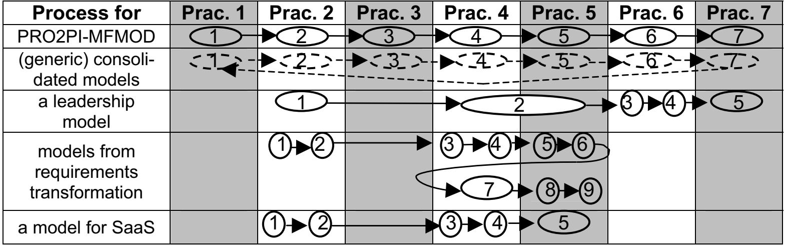
The process for a CMMI specialization to CBSE customizes the method framework applying the following customization rules: (a) rule CR2 is applied related with practice 1 because the initial decisions were already taken before the process was defined; (b) rule CR1 is applied four times because each one of the activities 1, 2, 3 and 4 corresponds to the practices 2, 3, 4 and 5; (c) rule CR5 is applied four times because each one of the activities 5, 6, 7 and 8 is more general and simple than the correspondents consecutives practices (3 and 4), (3 and 4 again), (2, 3, 4 and 5), and (6 and 7); (d) rule CR6 is applied three times because each one of the activities 5, 6, and 7 are cycles: activity 5 repeats practices 3 and 4, activity 6 repeats practices 3 and 4 again and activity 7 repeats practices 1, 2, 3 and 4.

The process for a CMMI specialization to banking domain customizes the method framework applying the following customization rules: (a) rule CR2 is applied related with practice 1 because the initial decisions were already taken before the process was defined; (b) rule CR1 is applied because the activity 1 corresponds to practice 2; (c) rule CR2 is applied for the no correspond activity for practice 3 because the strategy for the development (the result of practice 3) was already defined before the process; (d) rule CR4 is applied two times because each one of the consecutive activities (2 and 3) and (4 and 5) corresponds to the practices 4 and 5 respectively; and (e) rule CR1 is applied two times because each one of the activities 6 and 7 correspondents to practices 6 and 7 respectively.

The process for SPICE for Research customizes the method framework applying the following customization rules: (a) rule CR2 is applied related with practice 1 because the initial decisions were already taken before the process was defined; (b) rule CR4 is applied because the activities 1 and 2 are more detailed than the correspondent practice 2; (c) rule CR2 is applied for the no correspond activity for practice 3 because the strategy for the development (the result of practice 3) was already defined before the process; and (d) rule CR1 is applied four times because each one of the activities 3, 4, 5 and 6 correspondents to practices 4, 5, 6 and 7 respectively.

Table 2 shows the PRO2PI-MFMOD´s seven practices and the activities of each one of the four other processes described in Section 4 and indicate how each activity is related with the practices. For the generic process for consolidated models, we estimate a general process as cycles of PRO2PI-MFMOD´s seven activities.

Table 2 – Practices of PRO2PI-MFMOD and activities of four other processes



Process for	Prac. 1	Prac. 2	Prac. 3	Prac. 4	Prac. 5	Prac. 6	Prac. 7
PRO2PI-MFMOD	1	2	3	4	5	6	7
(generic) consolidated models	1	2	3	4	5	6	7
a leadership model		1		2		3 4	5
models from requirements transformation		1 2		3 4 7	5 6 8 9		
a model for SaaS		1 2		3 4	5		

7 Using the Method Framework for a Complex System

This session introduces the issues of application of this method framework for building process capability models in the context of complex systems. There is no consensus on the definition of complexity in the literature [18]. In the functionalist sense of the word, complexity refers to a large set of variables whose relations cannot be mapped or monitored [18]. For Demo [19], complexity is linked not only with the number of variables, but with a set of properties for interpreting a phenomenon as complex. The properties highlighted by the author are: the dynamics, the ambiguity, and the no-linearity. What is totally predictable and linear is not complex. These properties above help to characterize the complex a phenomenon as complex. The complex system in focus here is the Brazilian Public Software *(*SPB after the Portuguese name: *Software Publico Brasileiro)* [17].

The concept of public software in Brazil has its first public records of discussion in the 90's [17]. The first experiments supported conceptual nuances that had different scales, ranging from the software to be shared only in the public sector to the total release to society. In 1995 the state computing companies, captained by ABEP, began a process of discussion on what later became the concept of SPB [17]. At that time the intention was to accelerate cooperation in the government, in order to reduce developmental efforts, assign costs and rationalize resources. The trend for the total release of solutions to society is recent. Their format comes from the experience of the federal government.

A one year project is under way to consolidate a technical framework for SPB. One part of this project is a subproject to identify and consolidate, as process capability models, best practices for developing and evolve software or services and best practices to perform a service. This subproject has three sequential phases: (Phase 1) consolidation of this method framework and understanding of the SPB; (Phase 2) development of a draft version of the model; and (Phase 3) validation and consolidation of an initial version of the model. Phase 1 is already complete and Phases 2 and 3 are planned as an instantiation of this method framework.

This instantiation is composed of fourteen activities: (1) initial decisions; (2) sources identifications and initial analyses; (3) strategy for development; (4) detailed analyses of the identified sources; (5) detailed of the strategy; (6) high level model design; (7) revision of sources and new analyses; (8) revision of the strategy; (9) model design; (10) draft model development; (11) initial validation; (12) draft model development; (13) validation; (14) model consolidation. Table 3 shows the activities of this planned process and relate them with the practices of the method framework as applications of the customization rules.

Table 3 – Practice of PRO2PI-MFMOD and activities of a process for SPB complex system

Process for	Prac. 1	Prac. 2	Prac. 3	Prac. 4	Prac. 5	Prac. 6	Prac. 7
PRO2PI-MFMOD	1	2	3	4	5	6	7
SPB complex system	1	2 4 7	3 5 8	6 9	10 11	12 13	14

8 Initial Validation and Conclusion

Although this is a work in progress, the achievement of the three unfolded objective goals is commented as an initial validation. The achievement of Goal G1 is evidenced by Tables 1 and 2 showing that the activities of each one of the nine identified processes can be expressed with applications of the seven customizations rules on the seven PRO2PI-MFMOD´s practices. The achievement of Goal G2 is evidenced by Figure 1 showing PRO2PI-MFMOD as one element of PRO2PI methodology. Finally the achievement of Goal G3 is evidenced by Table 3 showing that the activities of the planned process for engineering a process capability model for SPB complex system can be expressed with applications of the seven customizations rules on the seven practices of PRO2PI-MFMOD.

This article introduced PRO2PI-MFMOD as a Method Framework for Engineering Process Capability Models. This method framework supports the definition of methods or processes to engineer a process capability model. The achievement of the three derived objective goals indicates a first confidence that PRO2PI-MFMOD is going to be a useful proposal for developing methods and processes for engineering process capability models.

Acknowledge

The authors acknowledge FINEP – *Financiadora de Estudos e Projetos* (Research and Projects Financing, also known as the Brazilian Innovation Agency) for finance support for this work under the project 0489/08: *Modelo de Referência para o Software Publico Brasileiro*.

Literature

1. Paulk, M. C., Weber, C., Curtis, B. and Chrissis, M. B., The Capability Maturity Model - Guidelines for Improving the Software Process, Addison-Wesley, 441 pages, 1994.
2. Chrissis, M. B., Konrad, M., Shrum, S., CMMI: Guidelines for Process Integration and Product Improvement, 2nd Edition, Addison-Wesley, 2007.
3. Salviano, C. F., “Model-Driven Process Capability Engineering for Knowledge Working Intensive Organization”, Proc. 8th Int. SPICE Conf., Nuremberg, Germany, May 2008, 9 pages.
4. Salviano, C. F. and Jino, M.: Towards a {(Process Capability Profile)-Driven (Process Engineering)} as an Evolution of Software Process Improvement, EuroSPI Industrial Proc., Finland, p. 12.26-12.37, 2006.
5. Salviano, C. F. and Figueiredo, A. M. C. M., “Unified Basic Concepts for Process Capability Models”, in Proc. 20th Int. Conf. on Sw. Eng. and Knowledge Eng. SEKE, San Francisco, USA, July 1-3, 2008, pp. 173-178.
6. The Int. Organization for Standardization and the Int. Electrotechnical Commission, ISO/IEC 15504 - Information Technology - Process Assessment, published as seven parts in different years: 15504-1:2004, 15504-2:2003, 15504-3:2003, 15504-4:2004, 15504-5:2006, 15504-6:2008 and 15504-7:2008.
7. Miranda, A. P. C., Uma abordagem com a ISO/IEC 15504 (SPICE) para melhoria do processo de ensino de cursos da área de informática em um Centro de Educação Profissional SENAC/GO, Monografia, Adviser: C. F. Salviano, Curso de especialização em Melhoria de Processo de Software, UFLA, Lavras, MG, 2005.
8. Anacleto, A., Gresse von Wangenheim, C., Salviano, C. F. and Savi, R., A Method for Process Assessment in Small Software Companies. 4th Int. SPICE Conf., Portugal, 2004.
9. Tsukumo, A. N., Santos, R. V. M., Marinho ,W., Silva, L. P., Salviano, C. F., Uma Estratégia para Melhoria de Processo de Desenvolvimento de Software Baseado em Componentes. V SBQS, Vila Velha, Brazil, 2006.
10. Cavalcante, Kelly V., e Costa, Roberta P. da, Método para Especialização de Modelos de Capacidade de Processo, Monografia, Adviser: C. F. Salviano, Faculdades Senac de Ciências Exatas e Tecnologia, curso de especialização em Qualidade no Desenvolvimento de Software, 2006.
11. Silva, J. V. L. da, Nabuco, O. F., Salviano, C. F., Reis, M. C., and Maciel F., R.: Strategic Management in University Research Laboratories Towards a Framework for Assessment and Improvement of R&D Management, Int. SPICE Conf. Proc., South Korea, 2007.
12. Silva, J. V. L. da, Desenvolvimento de um modelo para melhoria e avaliação da pesquisa em laboratórios universitários (in Portuguese: A model development for improvement and assessment of research in university laboratories), PdD thesis, Unicamp FEQ, 2007.
13. Tuffley, D., Evolving a Process Reference Model for the Leadership of Integrated Virtual Teams, Proc. 8th International SPICE Conference, Nuremberg, Germany, 26-28 May 2008.
14. Barafort, Béatrix, Renault, A., Picard, M., and Cortina, S., A Transformation Process for Building PRMs and PAMs based on a Collection of Requirements, Proc 8th Int SPICE Conf, Nuremberg, Ger., 26-28 May 2008.
15. Cancian, M. H., Uma Proposta de Guia de Referência para Provedores de Software como um Serviço, MSc Dissertation, UFSC, Automation and Systems Engineering, 2009.
16. Firesmith, D. G., with Capell, P., Hammons, C. B., Latimer, D., Merendino, T., The Method Framework for Engineering System Architectures, CRC Press, 2008.
17. Brazilian Public Software Portal, www.softwarepublico.gov.br/spb/ (in Portuguese).
18. Borgatti Neto, R., Perspectivas da Complexidade aplicadas à Gestão de Empresa. PhD Thesis, Escola Politécnica da USP, Dep. Engenharia de Produção, São Paulo, 2007.
19. Demo, P., Política Social do Conhecimento, Petrópolis, Vozes, 1999.

Author CVs

Clenio F. Salviano

Clenio F. Salviano works at CTI (*Centro de Tecnologia da Informação Renato Archer*) (www.cti.gov.br) since 1984 in Brazil and teaches Software Process Improvement (SPI) courses at industry and academia. At CTI, he coordinates a SPI Research, Development and Innovation technical group. He is member of ISO/IEC 15504 (SPICE) group and a SCAMPI CMMI Lead Appraiser. He helped more than 80 organizations to perform SPI programs. He published and presented more than 60 articles and tutorials in relevant conferences world-wide. He received his PhD at Unicamp in 2006 proposing PRO2PI Methodology as an evolution of current SPI (http://pro2pi.wikidot.com/, clenio.salviano@gmail.com).

Alessandra Zoucas

Alessandra Zoucas received her B.Sc. in Computer Engineer from the Universidade do Vale do Itajaí (UNIVALI). She is a master student of the postgraduate program in Applied Computer at UNIVALI. At the Federal Institute of Santa Catarina (IFSC), she is a professor for the Graduate Program in Management of Information Technology. She also works as a consultant in software process improvement and is an officially authorized Implementer and Assessor of the Brazilian Software Process Improvement Model MPS.BR. Contact her at UNIVALI, Rod. SC 407, Km 04, 88122-000 São José/SC, Brazil; alessandra.zoucas@gmail.com

Jorge V. L. Silva

Jorge is a researcher at CTI - Science and Technology Ministry since 1988 to present working in research and industry projects in the robotics and industrial automation field. Invited researcher at Industrial Automatic Institute (IAI/CSIC) Spain (1989), Invited professor of the specialization courses in robotics at UNIMEP (1992) and at Unicamp (1994). Head of Robotics Division at CTI (1992-1994). Head and founder of Three dimensional Technologies Division at CTI (1996 - present). Under his coordination, implanted a tight cooperation with industry and the most important universities in Brazil supporting at moment almost 30 PhD and Master´s thesis. His PhD was in the field of maturity models for research activities.

Angela M. Alves

Ph.D. in Information Technology by USP/POLI/Production (in process), Quality Master in the Unicamp / College of Engineering Mechanics (2002), Electrical Engineer / Electronic Barretos by the Foundation of Education (1977), Coordinator of postgraduate studies "lato sensu" Federal University of Lavras and researcher at CTI Renato Archer. Experience in the area of Software Engineering mainly working on software engineering, free software (OPENBRR, Software Público Brasileiro), acquisition process (eSCM, ACQ v1.2 CMMI, ITIL, CMMI- SVC), software process improvement (CMMI DEV v1.2, ISO/IEC15504) and corporate governance (COBIT). She meets the prerequisite to participate in official SCAMPI assessments. Address to access this CV: http://lattes.cnpq.br/258313424330764

Christiane Gresse von Wangenheim

Christiane Gresse von Wangenheim is a professor at the Universidade do Vale do Itajaí (UNIVALI) and the Graduate Program in Computer Science of the Federal University of Santa Catarina (UFSC). Her research interests are software process improvement and assessment.

Previously, she worked at the Fraunhofer Institute for Experimental Software Engineering. She received a PhD in Production Engineering at the Federal University of Santa Catarina (UFSC) Brazil and a PhD in Computer Science at the University of Kaiserslautern, Germany. She's also a PMP - Project Management Professional and Assessor of the Brazilian Process Improvement Model MPS.BR. Contact her at gresse@gmail.com.

Marcello Thiry

Marcello Thiry is a professor at Universidade do Vale do Itajaí (UNIVALI) since 1993 and consultant at Incremental Tecnologia since 2005. His research interests are software process improvement, including project management and requirements engineering. He received a PhD in Production Engineering at Federal University of Santa Catarina (UFSC) Brazil. He is also a PMP - Project Management Professional and Lead Assessor of the Brazilian Process Improvement Model MPS.BR. He is a member of the IEEE Computer Society and of the Project Management Institute. Contact him at UNIVALI, Rod. SC 407, Km 04, 88122-000 São José/SC, Brazil; marcello.thiry@gmail.com

Analysis of the Design of Software Process Assessment Methods from an Engineering Design Perspective

Mohammad Zarour, Alain Abran, Jean-Marc Desharnais

Abstract

Software process assessment (SPA) plays an important role in the Model-based software process improvement (SPI) paradigm; assessment methods are used to understand software organizations' current process quality and identify the possible improvement opportunities. This paper studies the design process of the SPA methods from an engineering viewpoint and uses Vincenti's classifications of engineering design knowledge as an analytical tool. The analyses end up with the necessary pieces of knowledge that the SPA methods' designers bring with them before starting the design process of the SPA method. These pieces of knowledge provide useful guidelines, mainly for less experienced designers, to start SPA methods design. Note that other design criteria can be induced in the same manner from the engineering design knowledge

Keywords

Software, process, assessment, improvement, design, engineering

1 Introduction

Software Process Assessment is an effective method used to understand software organizations' process quality and to identify issues to be resolved to achieve higher maturity [1]. In the past two decades, various assessment methods have been developed. These assessment methods varies from comprehensive SPA methods, such as SCAMPI method of CMMi [2] and SPA methods compliant with ISO 15504 [3], to lightweight assessment methods, see for example [4-9]. The effectiveness of any assessment method is affected by the size of the organization, i.e. for small and very small organizations, the comprehensive SPA methods are considered to be difficult to implement [10-14]. Accordingly, the SPA method effective for large organizations will not be effective for small or very small organizations.

Software processes assessment can be used either to determine the capability of another organization for subcontracting purposes, or to determine and understand the status of the organization's current processes to initiate an improvement process. The increasing numbers of assessment methods available, the ISO 15504 standard that sets out the requirements for process assessment, and the popularity of the CMMI model, illustrate the relevance of software process assessment for the software development industry. The increasing numbers of assessment methods encouraged several researchers to study the differences between various SPA methods and compare between them using different approaches - see for example [15-19].

The same requirements for conducting successful assessments are common to all SPA methods [1], The requirements consists of a set of high-level design criteria for developing, defining, and using assessment methods [20]. Usually, the design criteria for comprehensive assessment methods are well-defined, while for tailored lightweight methods the design of the assessment methods interferes with the designers' experience and opinion. Accordingly, identifying the design criteria to design an SPA method would help in standardizing the design process of comprehensive and lightweight SPA methods. Moreover, aligning the design knowledge of SPA methods (both comprehensive and lightweight methods) with the engineering design knowledge would help improve the maturity of the SPA methods' design. This paper studies the design knowledge of the SPA methods from an engineering viewpoint using Vincenti's classifications of engineering design knowledge as an analytical tool.

This paper presents in section 2 an overview of Vincenti's classifications of the engineering design knowledge. Section 3 presents the mapping of Vincenti's classifications of the engineering design knowledge to the SPA methods design. Section 4 summarizes the final results and presents the future work.

2 Vincenti's classifications

The work done by Vincenti [22] in defining the anatomy of engineering design knowledge based on a long experience in the aeronautical field forms a good framework to study the design process in the SPA field. Vincenti stated that "a complicated technology can often be regarded as a device". Today, the software products, which are used as stand alone products or as embedded in very complex systems, as well as the development process producing them, are obviously complex technologies and can be regarded as devices performing certain functions. Therefore, this paper uses Vincenti's classifications of the engineering design knowledge to study the SPA methods design from an engineering viewpoint.

Using Vincenti's terms and concepts in this SPA context, designing a new SPA method is mostly based on a vicarious model. The common vicarious models used in the SPA field are ISO 15504 and CMMI. Such a vicarious means of selection is preferred as a cost and time saving alternative of building a full assessment model and the corresponding assessment methods.

In his book [22], Vincenti discusses the anatomy of design knowledge in the engineering discipline and provides a categorization of engineering design knowledge. This categorization could also be used as an analytical tool to study the coverage of different engineering topics within other domains such as software engineering; for example, [21] presented the modeling of Vincenti's classifications and how to use Vincenti's categories as constituting criteria for investigating software engineering from an engineering perspective.

Accordingly, to investigate software process assessment methods using Vincenti's classifications, it is useful to understand to what extent the design of these methods aligns with engineering design principles.

Vincenti stated that this classification is not specific to the aeronautical engineering domain only, but can be transferred to other engineering domains. This transfer to the software engineering field in general and software process assessment and improvement in particular, is challenging in the sense that this field is still maturing. The six main classes of Vincenti's classification are:

1. Fundamental design principles
2. Criteria and specifications
3. Theoretical tools
4. Quantitative data
5. Practical considerations
6. Instrumentalities

3 Mapping Vincenti's classification to SPA methods design

This section uses Vincenti's classification as a tool to analyse the design process of the SPA methods, and map the design criteria in the SPA context to the engineering design classifications.

3.1 Fundamental design principles

Usually, the designers planning to start a project to build a certain device using a normal design process bring with them some fundamental concepts about the devices. These concepts may exist only in the designers mind implicitly or stated explicitly somewhere else: "they are givens for the projects, even if unstated" [22]. As stated by Vincenti, the fundamental design concepts can be derived from two main sources:

3.1.1 *Operational principles*

These principles specify how the different parts of the designed device fulfill special functions in combination with overall operation to achieve the purpose. In other words "how the device works. The operational principles also, in effect, define a device" [22].

The main principle to design an SPA method – the proposed device – is that the designer keeps in mind that the software development process should be divided into a set of distinct processes. For each process a clear definition of purpose and outcomes is provided; this is formally known as a process reference model. Consequently, the design criterion corresponding to operational principles is:

1. Identify the process reference model.

3.1.2 *Normal configuration*

The normal configuration of a device means “the general shape and arrangement that are commonly agreed to best embody the operational principles” [22]; that is to say, any device or product to be produced, usually, consists of a set of sub-devices or sub-products, the interaction and arrangement of these sub-products is what concerns a normal configuration.

Accordingly, the design criteria corresponding to the configuration management include:

1. Define the business need before the assessment.
2. Make use of previous assessment reports.
3. Refer to the organizational documents and reports while preparing for the assessment.

3.2 Criteria and specifications

Vincenti stated that “to design a device embodying a given operational principle and normal configuration, the designer must have at some point specific requirements in terms of hardware”. When designing a new device, the designer translates the general qualitative goals into specific quantitative goals. The designer must have knowledge of technical criteria appropriate to the device and its use; the designer must also assign numerical values or limits to the characteristics of the appropriate criteria which is essential for the design. Therefore, the design criteria for SPA methods include:

1. Specify the number of processes to be assessed.

2. Specify the criteria for assessing the process.

3. Define the scale and its limits used to assess the process.

3.3 Theoretical tools

Vincenti stated that "To carry out their design function, engineers use a wide range of theoretical tools. These include intellectual concepts for thinking about the design as well as mathematical methods, theories and formulas which can be simple or complex formulas for making design calculations" [22]. Accordingly, the design criteria for this class include:

1. Specify the theoretical tools used to select the processes to be assessed.

2. Specify the theoretical tools used to define the rating process.

3.4 Quantitative data

Vincenti focused on the importance of quantities and other data for other physical properties required in the formulas during the design process. Vincenti also stated that “other kinds of data may also be needed to lay out details of the device or to specify manufacturing processes for production” [22]. Such data is usually obtained empirically and sometimes calculated theoretically and are typically represented in tables or graphs. The quantitative data can be divided into two types of knowledge, descriptive and prescriptive [22].

Descriptive knowledge is the knowledge of how things are. It includes physical constants as well as properties of substances and physical processes. Descriptive data occasionally deal with operational conditions in the physical world.

Prescriptive knowledge is knowledge of how things should be to attain a desired end – it says, in effect, “in order to accomplish this, arrange things this way” [22].

The design criterion related to this class is:

1. What data/indicators are used to determine the scale for each process?

3.5 Practical considerations

In addition to theoretical tools and quantitative data, Vincenti stated that “Designers also need for their work an array of less sharply defined considerations derived from experience in practice” [22]. Usually, practical considerations are difficult to define and are rarely documented. Sometimes the practical considerations become well codified. In such cases, these practical considerations are moved to another category.

When designing the assessment method, the designers specify whether to include all process from the reference model or select a set of processes to be assessed either based on their own experience or by applying certain selection methods. When rating the organization with reference to an assessment model, designers also specify the maximum target scaling level to be used based on their experience and according to the needs of the organization. Designers also should decide whether to build an action plan or not at the end of the implementation of the assessment method. Hence, the related design criteria include:

1. How the processes to be assessed are selected?

2. What is the target scaling level for the organization?

3.6 Instrumentalities

“Besides the analytical tools, quantitative data and practical considerations required for their tasks, designers need to know how to carry out those tasks” [22]. As part of the engineering design knowledge, the instrumentalities of the design process should be specified which contain the procedures, judgmental skills and ways of thinking by which the process is done.

Vincenti mentioned that “designers doing normal design call upon a number of well-organized, more or less structured procedures”; Vincenti also mentioned that “the division of an overall system into subsystems is fundamental. In the terms of the SPA method, the assessment process is divided into subdivisions, or phases using Loon’s terms [23]. The arrangement and configuration of each sub division is defined by the designer: these sub divisions are executed sequentially when conducting the assessment method which defines the assessment procedure. Examples of such sub-division are the assessment phases defined by Loon as discussed previously. These divisions may vary from one assessment method to another. Consequently, the design criterion related to this issue is:

1. Define the sub divisions of the assessment method during the assessment design process.

4 Summary and future work

This paper studied the design criteria of SPA methods from an engineering viewpoint. Aligning the design of SPA methods with the engineering design knowledge helps improving the maturity of software engineering field in general as an engineering discipline. The resulted design criteria are classified based on Vincenti’s classifications of engineering design knowledge.

During the design phase of the SPA method, the designer should take these criteria into consideration. Failing to take one or more of these criteria into consideration would be considered a weakness point in the design process that may cause an ineffective implementation of the SPA method.

In this paper, we have studied the different design criteria of SPA methods from an engineering viewpoint based on Vincenti’s classifications. Vincenti’s classification consists of six main classes that cover the different aspects of the engineering design process. The resulted design criteria are vital for designing SPA methods and can be used by designers of new SPA methods as guidelines to direct the design process of the new SPA method.

Future work is in progress and aimed at defining other design criteria using the same approach presented in this paper, and also developing an evaluation method, based on the identified design criteria, to evaluate the success level of the designed SPA methods.

Literature

[1] T. Komiyama, T. Sunazuka, and S. Koyama, "Proposal on Library–Centered Software Process Assessment," *CrossTalk - The Journal of Defense Software Engineering,* vol. 14, pp. 22-28, August, 2001 2001.

[2] CMMI Product Team, *CMMI for Development Version 1.2*. Pittsburgh, PA: Carnegie Melon, Software Engineering Institute, 2006.

[3] ISO/IEC, "ISO/IEC 15504 Information Technology - Process Assessment - Parts 1-5," 2003-2006.

[4] G. A. Cignoni, "Rapid Software Process Assessment to promote Innovation in SMEs," in *EUROMICRO'99*, Milan, Italy, 8-10, September 1999, 1999.

[5] C. Wangenheim, A. Anacleto, and C. F. Salviano, " MARES - A Methodology for Software Process Assessment in Small Software Companies, Technical Report: LQPS001_04E," LPQS - Universidade do Vale do Itajai, Brazil, Technical Report 2004.

[6] T. P. Rout, A. Tuffley, B. Cahill, and B. Hodgen, "The Rapid Assessment of Software Process Capability," in *First International SPICE Conference*, Limerick, Ireland, 2000.

[7] N. Habra, A. Renault, S. Alexandre, and M. Lopez, "OWPL Micro Assessment," in *Software Quality Workshop, 24rd International confernce on Software Engineering ICSE*, Orlando, Florida USA, 2002.

[8] A. Beitz, K. E. Emam, and J. Jarvinen, "A business focus to assessments," in *European Conference on Software Process Improvement*, 1999, p. 6.

[9] F. McCaffery, D. McFall, and F. G. Wilkie, "Improving the Express Process Appraisal Method," *PROFES,* pp. 286-298, 2005.

[10] A. Laryd and T. Orci, "Dynamic CMM for Small Organizations," in *First Argentine Symposium on Software Engineering - ASSE 2000*, Tandil, Argentina, 2000, pp. 133-149.

[11] D. L. Johnson and J. G. Brodman, "Tailoring the CMM for Small Businesses, Small Organizations, and Small Projets," in *Software Process Newsletter*. vol. 8: IEEE Computer Society, 1997.

[12] D. P. Kelly and B. Culleton, "Process Improvement for Small Organization," *Computer,* vol. 32, pp. 41-47, Oct. 1999 1999.

[13] C. Laporte and A. April, "Applying Software Engineering Standards in Small Settings: Recent historical perspectives initial achievements," in *International Research Workshop for Process Improvement in Small Settings*, Software Engineering Institute, Pittsburgh, 2005.

[14] J. A. Villalón, A. G. Cuevas, G. T. San Feliu, S. A. De Amescua, S. L. García, and C. M. Pérez, "Experiences in the Application of Software Process Improvement in SMES," *Software Quality Journal,* vol. 10, pp. 261-273, November 2002.

[15] M. O. Tingey, *Comparing ISO 9000, Malcolm Baldrige, and the SEI CMM for software: a reference and selection guide*: Upper Saddle River: Prentice-Hall, Inc., 1997.

[16] K. El-Emam, J.-N. Drouin, and M. Walcélio, *SPICE - The Theory and Practice of Software Process Improvement and Capability Determination*. Los Alamitos CA: Wiley-IEEE Computer Society Press, 1998.

[17] M. C. Paulk, "How ISO 9001 Compares with the CMM," *IEEE Software,* vol. 12, pp. 74-82, Jan, 1995 1995.

[18] C. Halvorsen and C. Reidar, "A Taxonomy to Compare SPI Frameworks," in *Lecture Notes In Computer Science; Vol. 2077, Proceedings of the 8th European Workshop on Software Process Technology* Witten, Germany, 2001, pp. 217 - 235.

[19] M. Zarour, J.-M. Desharnais, and A. Abran, "A Framework to Compare Software Process Assessment Methods Dedicated to Small and Very Small Organizations," in *International Conference on Software Quality - ICSQ'07* Denver, CO, USA, 2007.

[20] CMMI-SUT, "CMU/SEI-2006-TR-011: Appraisal Requirements for CMMI, Version 1.2 (ARC, V1.2), SCAMPI Upgrade Team," Carnegie Mellon Software Engineering Institute, Pittsburgh, PA CMU/SEI-2006-TR-011, August 2006 2006.

[21] A. Abran and K. Meridji, "Analysis of Software Engineering from An Engineering Perspective," in *UPGRADE*. vol. VII, 2006.

[22] W. G. Vincenti, *What engineers know and how they know it*. Baltimore, London: The John Hopkins University Press, 1990.

[23] H. v. Loon, A. Cass, and J. c. Linde, *Process Assessment and Improvement: Apracticle guide for managers, quality professional and assessors* vol. 776. NY, USA: Springer, 2004.

4 Author CVs

Mohammad Zarour

Dr. Mohammad Zarour holds a Ph.D. in Software Engineering (2009) from École de Technologie Supérieure (ETS) (Montreal, Canada) and master degree in computer science (1998) from University of Jordan (Amman, Jordan).

He is currently a Professor at the University of Petra (Amman, Jordan). He has 8 years of experience in teaching in a university environment as well as more than 2 years of industry experience in information systems development and software engineering. His research interests include software quality, software process assessment and improvement, and software measurement. Dr. Zarour is a Canadian representative at ISO SC/7 WG-24 (Lifecycle profiles for very small organizations).

Alain Abran

Dr. Alain Abran holds a Ph.D. in Electrical and Computer Engineering (1994) from École Polytechnique de Montréal (Canada) and master degrees in Management Sciences (1974) and Electrical Engineering (1975) from University of Ottawa.

He is a Professor and the Director of the Software Engineering Research Laboratory at the École de Technologie Supérieure (ETS) – Université du Québec (Montréal, Canada). He has 15 years of experience in teaching in a university environment as well as more than 20 years of industry experience in information systems development and software engineering. His research interests include software productivity and estimation models, software engineering foundations, software quality, software functional size measurement, software risk management and software maintenance management.

Dr. Abran is currently a co-editor of the Guide to the Software Engineering Body of Knowledge project and a Co-chair of the Common Software Measurement International Consortium (COSMIC).

Jean-Marc Desharnais

Dr. Jean Marc Desharnais have been working in the computer and software engineering fields for more than 30 years, as an analyst and project manager at the start of his career, and, most recently, as a software engineering consultant and professor. Since 1987, the expertise he has developed has mainly been in project management, specifically software measures for business applications. He is the Canadian representative at ISO SC/7 WG-6 (Quality measures) and the editor of the Quality Measure Element standard (ISO/IEC 25021). Dr. Desharnais is also co-founder of COSMIC in 1998 and co-author of the COSMIC functional size measurement method. He taught for 4 years as full time professor at ÉTS (École de Technologie Supérieure) in Montreal, Canada and since the fall of 2008 as Lecturer at Middle East Technical University (METU) in Ankara, Turkey. He has close to a hundred publications (see www.gelog.etsmtl.ca), and is doing consulting work across the world.

Experiment with COSMIC V3.0: Case Studies in Business Applications

Sanae Saadaoui, Annick Majchrowski and Christophe Ponsard
CETIC, Charleroi, Belgium
{sanae.saadaoui, annick.majchrowski, christophe.ponsard}@cetic.be
http://www.cetic.be

Abstract

Software measurement currently plays a crucial role in software engineering. It helps analysing both quality and productivity of the developed or maintained software. COSMIC–FFP, the COmmon Software Measurement International Consortium Full Functional Point (CFFP) is a measurement methodology designed to estimate the functional size of software. While bundled with comprehensive guidelines, the practical application of COSMIC is not trivial and experience plays an important role for its successful application. The objective of this paper is to provide practitioners with hints and additional recommendations learnt from applying CFFP to significant industrial cases. More specifically this paper will focus on the latest COSMIC version (COSMIC V3.0) which comes with new concepts and for which there is still not much experience returns from its application. This paper relies on two significant case studies covering different domains with different kinds of measurements: the first one being a wide-scale web application for a European Commission administration to be significantly improved with the measurement of all enhancements requirements, the second one being a modular application to be used jointly by five Belgian federal and regional parliaments with the measurement of initial requirements before development.

Keywords

Functional Size Measurement, COSMIC V3.0, modular application, business application

1 Introduction

The COSMIC method is one of the largely used methods of sizing software estimation [2][4]. The interest in COSMIC comes from the fact that it is based on objective criteria and rules which allows repetitiveness of measures. The method is innovative [8][9] because it is applicable early in the software lifecycle, already at specifications level. Another main advantage of the CFFP is its complete independence of software development technologies and methods. The recent versions of the method are largely generic and cover many software domains from business applications [5] to real-time systems [12].

The aim of this paper is to present a set of practical recommendations that we hope may help COSMIC V3.0 practitioners. These recommendations are based on typical problems encountered when applying COSMIC V3.0 in representative case studies. The present work addresses in particular the impact of the choice of documents and artefacts on the COSMIC V3.0 estimation process and on the effort needed to perform it. Some among the problems discussed are:

- the choice of artefacts and documents to use for software size estimation
- the choice of the level of specification details, especially when artefacts have different granularity levels
- the impact of the structure of the document in particular on the mapping

All those may have great impact on the estimation effort. Although already partially addressed by guidelines such as [5], we give here a more practical feedback based on two case studies related to large systems. The first one concerns the estimation of the size of a change to a large scale web application for a European Commission administration. The second case study is a size estimation of a new modular access card system to connect five Belgian parliaments.

The COSMIC V3.0 estimation was achieved at the end of the specification analysis phase. Both projects cited above are not entered the implementation phase. The present work will be completed by a comparison of the estimation results with real data after the software is implemented.

The present paper only considers the business applications. However, it presents observations bound to a large spectrum of contexts: measurement of new development software, measurement of enhancement development software and government procurement procedure.

The paper is organised as follows: section 2 presents an overview of the COSMIC model V3.0. Sections 3 and 4 respectively describe two case studies using a similar structure: a first sub-section is dedicated to the context and technical aspects of the application. It is followed by a second sub-section giving highlights of how COSMIC was applied. This sub-section focuses only on key and relevant aspects and differs from one case study to the other. Section 5 summarises lessons learnt from the experiments and gives some practical recommendations for future applications. Section 6 gives some conclusions and future work directions.

2 Overview of the COSMIC V3.0 method

The COSMIC-FFP [2][4][5][6] is a standardised method – ISO/IEC 19761:2003 - designed for measuring the functional size of software [4][5][6]. The method is applicable for data rich applications: Business application software, real-time software and hybrid application of the above. These types of software are characterised by functions that input data, store and retrieve data, and output data. The COSMIC method is not applicable to size estimation of software dominated by functions that mainly manipulate data as in typical scientific software. The measurement can be carried out at any phase of the life cycle of the software, whether the software already exists or not. The COSMIC method measures a size as seen by the “functional users” of the piece of software to be measured. The method uses a model based on the following principles:

- **Functional User Requirements (FUR)** of a piece of software which are composed by a set of unique functional processes.
- Each **functional process** is triggered by an ‘Entry’ data movement from a functional user which informs the functional process that the functional user has identified an event that the software must respond to.
- A data movement moves a single **data group** of attributes describing a single ‘object of interest’.
- There are four types of **data movement**. An ‘Entry’ moves a data group into the software from a functional user and an ‘Exit’ moves a data group out. ‘Writes’ and ‘Reads’ move data group to and from persistent storage, respectively.

The size of a piece of software is then defined as the total number of data movements (Entries, Exits, Reads and Writes) summed over all functional processes of the piece of software. Each data movement is counted as one ‘COSMIC Function Point’ of (‘CFP’), CFP being the unit of COSMIC measurement.

The COSMIC V3.0 method is applied in three phases: the strategy phase, the mapping phase and the measurement phase. The workflow of the method is illustrated in the following figure (Figure 1).

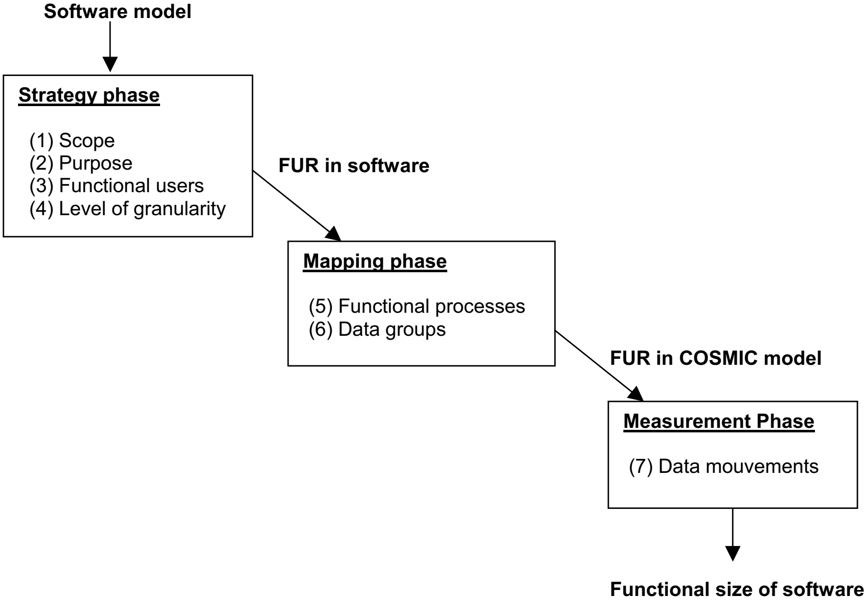


Figure 1: COSMIC V3.0 workflow method

The **Strategy phase** consists in defining the context of the measurement: the answers to the questions "what", "why" and "how" to measure are given in this phase. Four key COSMIC concepts are thus introduced: (1) the scope (to respond to "what" is to be measured) which allows to identify the set of FUR to measure) (2) the purpose of the measurement (to respond to "why" measure) (3) the functional users who will use the pieces of software; users may be humans or peer applications (4) the level of granularity (to respond to "how") which specifies the level of details at which the measurement should be carried out.

The **Mapping phase** consists in mapping the software to the COSMIC model. FUR of the software to measure is analysed: the set of triggering events sensed by the functional users are identified, the set of corresponding functional processes (5) triggered by events are extracted from the FUR, the data groups (6) moving inside the functional processes are identified.

The **Measurement phase** consists in measuring each of the functional processes identified during the previous phase. For each functional process, the data movements (7) are identified. A data movement can be one of four types: an *ENTRY* (E) type moving a data group from a functional user to the functional process, an *EXIT* (X) type moving data from the functional process to the functional user, a *READ* (R) type moving a data group from a persistent storage to the functional process and a *WRITE* (W) type moving data from the functional process to a persistent storage. Next, the COSMIC measurement function is applied to each data movement. Each data movement is allocated 1 CFP (COSMIC Function Point). The functional size of a functional process is the sum of sizes of its constituent data movements. The size of the measured piece of software is the sum of the sizes of its constituent functional processes. Once the key concepts of the method are defined and described, the application of the method is straightforward and easy to carry out phase per phase. Unfortunately, decisions on how to break-up the software pieces into pieces if necessary, at which specification detail to work, and what level of granularity to use depend strongly on the software analysis skills and experience of the estimator. A careful analysis should be carried out to deeply understand the application to be measured. Often, the size estimation is carried out to provide an objective estimation of the development effort necessary to develop the estimated software. This is not part of the COSMIC method. However, it is possible to derive the needed numbers based on static tools such the International Software Benchmarking Standards Group, ISBSG, [14]. The repository contains a big number of carefully selected projects of different types and sizes. Relevant information on the development cycle of the projects is provided. The estimation of the development effort is given in real-time and in functional size.

Based on information from the ISBSG and on the collected specific data related to the development cycle of the company, a productivity rate is inferred. This rate allows one to calculate an estimation of the necessary effort per CFP.

3 Case Study 1 : Changes on a business application

3.1 Project description

The project is realised in a consulting software development context where the client is DGTaxud – the General Direction of Taxation and Customs Union, one of the administrations of the European Commission. The software development team is SBS - Siemens Business Services. Contact modalities require size estimation based on COSMIC and this estimation was contracted out to third party experts: CETIC – Centre of Excellence in Technologies of Information and Communication, an independent research centre.

The purpose of this project was to implement some change requests on a Service Management tool application supporting the essential processes included in ITIL (Information Technology Infrastructure Library) framework [13]. The changes to develop on this Web-based application are registered in a context of evolutionary maintenance of the application.

Two of the eleven ITIL processes are impacted by the changes. These are: (1) The Incident Management Process, which is responsible for the management of all incidents (corresponding to calls) addressed by users to Service Desk management and (2) The Report Management Process, which is responsible for the management of all reports created by Report Managers and executed by Report Executors. For every Report, Report managers define access rights for the Report Executors.

The changes to be developed are (1) Changes concerning the creation of an incident. In the new incident creation screen, four new fields are added to support four new objects which also have an impact (modification and/or deletion) on other existing items including changes to their functionalities. And (2) changes concerning reporting management. Templates of the different reports and export files have to be updated to take into account the new objects.

3.2 COSMIC application highlights

This section presents how COSMIC was applied and describes some key concepts as “*level of granularity*” and “*functional users*”, “*functional processes*”, “*data groups*” and “*data movements*” . We have focused on difficulties and facilities encountered when applying them.

Identification of the FUR

Before applying the COSMIC method, we have to identify the Functional User Requirements (FUR) of the piece of software to be measured. Two input artefacts were used for extracting the FUR. The first describes the changes on the Incident Management process and the second on the Report management Process. Extracting FUR is a crucial task which was very hard to perform due to the scope of the changes on the other ITIL processes. New impacted changes were analysed using relative artefacts and initial artefacts before changes. As suggested by COSMIC V3.0 [5], to measure changes, we first measure the existing functional processes. Secondly, we check how they will be functionally changed. Finally, the changes are measured.

Thus, an **important number of input artefacts** (five large artefacts) were used. They are written in **different forms** for **different kinds of readers** (end-users, analysts or developers). The two documents relating changes are mostly *developer-dedicated*. They are composed of a set of existing screen forms with their fields (and all possible values) and the detailed related changes - new, modified, deleted fields -. They also describe changes related to processed functionalities induced by these fields and some technical considerations. The third used document relating the specifications of the existing application are rather *analyst-dedicated* and contains a global description of all supported ITIL processes. The two last used documents are *end-users-dedicated* and contain a more detailed description of the functionalities for all ITIL data processes manipulated by the application. Note that all these differences between documents have contributed to complicate the extraction of the existing FUR and changed ones.

Strategy phase

Once having identified the FUR, we defined the real context of our measure. The four key concepts of this phase being the purpose, the scope, the functional users and the level of granularity of the measurement. These had to be identified before starting the measure.

The **functional users** defined as *"the senders or the intended recipient of data in the FUR"* [4] were easily identified. In V3.0, no difference is made between developer and end-user viewpoints as in version 2.2. [7]. Distinguishing between the two viewpoints in our case would have been difficult due to the large amount of available documentation.

The **level of granularity** is defined as "*any level of expansion of the description such that at each increased level of expansion, the description of the functionality is at an increased and uniform level of details*" [4]. In applying the definition, we found that the level of details of the documentation was strongly linked to the COSMIC concept of level of granularity: the highest level of details corresponding to the lowest levels of granularity. Since the available input artefacts were dedicated to different readers, functionalities were described at different levels of details: developer-dedicated documents being more detailed than end-user documents. In such situations, COSMIC V3.0 says that "*The recommended and unambiguous way to measure is to make the measurement with the same level of granularity and ideally with the level of granularity at which the functional processes have been identified"* [4]. The level of details in the used changes artefacts corresponds to a **functional sub-process level of granularity**. Additionally, as the scope was to measure the changes of a piece of software, we granted more importance to the level of details of changes than of existing functions. These were our motivations to **choose the lowest level of granularity** for our measurement.

Mapping phase

In this phase, the identified FUR is mapped to the COSMIC concepts such as "*functional processes"*, "*functional sub-processes"* and "*data groups"*. Focus was put on the easiness in deriving functional processes and data groups from the available artefacts.

To identify the **functional processes**, we followed one of the approaches described in [5] stating that *"It is sometimes helpful to first break down the FUR into their elementary parts or the smallest units of activity that are meaningful to the users (for example: definitions of screen, or report layouts)"*. In our case, the artefact relating changes for Incident Management was precisely composed of a set of screens and report layouts. We observed that defining **screens and reports layout** in the input documentation is a good practice to identify the functional processes.

To identify the **data groups**, we have not applied a data analysis method, as recommended [5]. We derived data groups by interpreting the fields and the columns as data groups. We observed that documentation containing directly the COSMIC concepts remains naturally the most interesting approach.

Measurement phase

In the context of a software evolution, we used the following approach to identify the **data movements** of each functional process: (1) consider the old situation and measure the data movements for the old functional process, (2) consider the new situation and measure the data movements by noting the changes (Added, Modified, Deleted, No change), and (3) the total functional size of the changed functional processes equals the sum of the data movements of all functional processes different from "No change". The following table illustrates the measurement of one changed functional process in the existing application that will sort all types (old ones and new asked ones) of incidents. The last column is used to facilitate the computation of only the changed movements (A/M/D). An empty column means that no change occurs.

Data movement	Number of data movements	Data groups movement description	Change type (A/M/D)
E	1	Sort asked by the functional user	
R	7	Existing incidents data : Call ID, organization, issuer, open date, close date, status, priority	
R	3	New incidents data: Business Threads, categories, configuration Items	A
X	1	List of sorted incidents	M
Total CFP	12		

The total size of this changed functional process is 4 CFP (only the changed data movements): 3 Read corresponding to the reading of the three new data incidents (in the corresponding data bases) + 1 Exit corresponding to the output of the required sorted list that is changed to support the new data incidents. No writing in data bases is necessary for this process.

4 Case study 2 : A modular business application, the PACXS project

4.1 Project description

The PACXS (Parliamentary Access Control eXchange System) is a data exchange system among five existing Belgian parliaments' access control systems. The project objective is to analyse the possibility to unify these access control systems. The application should guarantee a better secured access to all assemblies infrastructure. It should also ease the administration and improve usability due to the large number of people involved in several assemblies currently requiring multiple credentials. This requires fined-grained sharing of information of access control between those assemblies. It should also integrate itself with existing access control systems.

The application to develop had to be distributed into different logical modules grouping a number of components composed of a set of related functionalities. The main modules are: the access management system, the badge management system and the exchange management system. The modularity aspect of the application was central to the project to allow assemblies to continue to work separately and independently while integrating the required modules for a reliable exchange of information. The system is designed around a number of mandatory modules, beyond those; each assembly is free to integrate additional shared modules depending on her needs and existing infrastructure. This can support two kinds of deployment: a minimal one just relying on the core (targeting parliament with a full system in place) and a complete system (targeting parliaments with a partial system in place). The estimation is done for both kinds of modules with a clear interface on the interaction with the existing systems. The elaboration of the functional and technical description of the application went through the following steps:

- A functional requirements document describing the high level functionalities of the system using a goal-based approach [11]
- A Use case document describing business scenarios (between the user and the system) and more technical scenarios (at message exchange level), based on a UML approach
- A user interface document describing screen mock-ups
- A component model document describing component functional interfaces and interactions
- A detailed technical analysis describing all the functionalities of the system and messages

These analysis phases were achieved in a joint work by NSI and respect-IT under the project and quality supervision of the CETIC. The implementation is not part of the actual project. The possible implementation would be part of another public procurement procedure.

4.2 COSMIC application highlights

The highlights presented here cover mainly the choice of artefacts from which to derive key concepts like Functional User Requirement (FUR), functional processes and data groups and data movements.

Purpose of the measurement

The purpose of this measurement is to estimate the development effort for the development of the new modular application PACXS. The firm responsible for the development of the system will be selected through a Government Procurement procedure. The overall objective of this measurement is indeed to help choosing the adequate development firm.

Identification of Functional User Requirements

Functional User requirements are a set of user requirements that explains what the software will perform in term of tasks and services. The FUR was extracted from the specification document. The FUR is composed of all requirements on business cases and all those relative to message exchanges. The non-functional requirements are not part of the FUR.

Scope of the measurement

The singularity of the application to be developed is its modularity. Estimating the effort for the whole system as one piece of software was not an option it was explicitly required to produce module level estimations for the following reasons:

- to estimate the distribution of development costs between partners in a mutualisation perspective
- to produce several partial call for tenders
- to some extent: to identify the integration effort of existing modules in some parliaments

Choice of the artefacts

COSMIC estimation needs to define concepts at the logical level. In first place, we used the requirements document for a preliminary estimation. The document is well structured because it relies on goal-orientated requirements. It contains both functional and non-functional requirements. Only functional requirements are measurable by COSMIC. Even though complete in a functional point of view, the estimation was not satisfactory because such functionalities are described at too high level and do not allow to uncover all functional processes. For this purpose, the use case document which relates all business diagrams scenarios was used. However it did not provide detailed enough information on the messages exchanged. Hence for these aspects, the most detailed technical document was used.

Identification of functional users

The PACXS system is located at the architectural layer. Two kinds of users were identified. (1) Human users - technical staff responsible for the management and administration of the system – and (2) non human users which are the other modules exchanging information with the estimated one and external applications - local system of each assembly, badge provider application and local human resource applications. Those were easily identified at the requirement level document because those goal-oriented documents have a clear description of the responsibility assignments.

Level of granularity and functional processes identification

The FUR of software may be expressed at different levels of granularity. From more general specifications to more detailed requirements. For an accurate measurement, COSMIC requires measurements to be done at the functional process level. At higher levels, the results are approximations [1][4][5].

The PACXS measurement was performed at the functional process level. The particularity of this system is that it is composed of the access card management sub-system and the exchange management sub-system. For the first sub-system, the functional processes describe the different business scenarios from the human user point of view. For the second one, it was difficult to define functional process level without referring to technical specifications. It is well known that the functional size of software typically increases when going from a high granularity level to a lower one. A well established level of granularity at which to estimate regularly is crucial for software management and improvement. It was quite challenging to find a common level of granularity to apply COSMIC to all components because the work was carried out by two different people with quite different expertise (analyst vs. technical). An identified improvement here is to force the same level of granularity using common templates and possibly, earlier monitoring by the estimation team.

PACXS size measurement and development effort estimation

In the measurement phase, each module was sized by adding the data movements of all its functional processes. The following table illustrates the measurement of one functional process in PACXS system that will allow viewing the list of a given type of received messages.

Data movement type	Number of data movements	Data Groups
E	1	The type of message
R	2	DB_Message et DB_Badge
W	0	
X	1	List of messages of given type
Total CFP	4	

The total size of this functional process is 4 CFP : 1 Entry corresponding to the message type given by the functional user to the software application + 2 Read corresponding to the reading of message and access cards information (in the corresponding data bases) + 1 Exit corresponding to the output of the required list appeared on the screen. No writing in data bases is necessary for this process.

Each assembly can then compute the effort of its own development by considering only the modules it is interested in. Mutualisation can also occur to share costs. It is important to notice that the functional size of the whole PACXS system is different from the added sizes of all modules measured separately. Sizing the modules separately takes into account more interactions and then more data movements. To produce development efforts from the functional size, some kind of productivity coefficient is required. As no historical data was available for this specific context, the approach was to calculate the rate factor for effort estimation based on statistical data from the ISBSG repository [3]. An advantage of applying the estimation at module level is that it allows to apply different rates depending on the nature of the modules considered. Another issue requiring more investigation is the “novelty” and “regularity” aspects: some modules are completely new and very specific (business process level) while other are based on well known techniques and show systematic patterns. Hence, the productivity for the later modules is expected to be higher than the former ones.

5 Lessons learnt and some recommendations

This section presents the summary of the lessons learnt based on the experiment of applying COSMIC V3.0 on the two proposed case studies:

- Large amounts of artefact may be not productive. Fewer ones but highly and homogeneously structured at process level are more relevant for COSMIC measurement
- Identifying the FUR is not a trivial operation and depends on the provided artefacts. It is recommended to use carefully structured documents. A well structured document is one that (1) distinguishes between functional and non functional requirements; (2) describes the links from high level specification to lowest level ones and (3) highlights the inputs and outputs related to each specification. When producing such a document it is necessary to get specification de-

tails grouped in the same section to avoid redundancy when measuring. Goal-oriented requirements templates combined with use-case documents proved efficient in this.

- Mapping the FUR from the artefacts to the COSMIC concepts is not a trivial operation. A more accurate template of artefacts would be profitable to reduce mapping time. It would also provide a more objective interpretation method to derive COSMIC key concepts from the input document elements. These templates could be structured in sections correlated with expected COSMIC concepts. Such templates are currently being elaborated.
- Input artefact containing screens and report layouts facilitate in a significant way the identification of the functional processes and sub-processes during the mapping phase.
- An accurate size measurement depends closely on the way the strategic phase was conducted. The results of the measurements are strongly impacted by the purpose, scope and level of granularity. Different measurement contexts lead to different functional sizes. It is strongly advised to conduct this phase very carefully.

Defining an accurate level of granularity is a key success for defining a development effort methodology and an efficient effort estimation model.

6 Conclusion

In this paper we presented some lessons learnt together with contextualised and practical recommendations collected when applying the COSMIC V3.0 method on two important case studies. The observations show that analysing documents to extract the FUR and level of granularity is very demanding and time consuming. A well structured standard template for requirement documents may save time and reduce considerably the strategic phase. A common document helps defining a standard level of granularity and then defining a strategy for software size comparison and the establishment of an effort estimation model. The result is an improvement in the software development process. We are actually working on elaborating a template. We hope practitioners will find this set of recommendations useful for their own day-to-day work with COSMIC. The work will be continuously improved by current and future estimations. The future work goal is to define and validate a practical methodology on the application of the COSMIC V3.0 method.

Acknowledgement

This research project is supported by the FEDER - European Union and the Walloon Region under the terms defined in the Convention n°ECV12020011606F.

Literature

[1] Charles Simons, Advancing Functional Size Measurement – which size should we measure? Software Measurement European forum (SMEF 2007), Roma, Italy, May 9-11, 2007

[2] Alan Abran, Peter Fagg, Roberto Meli, Charles Simon, ISO transportation and Clarifications of the COSMIC FFP Method f Functional Sizing, IWSM 2002

[3] ISBSG. The International Software Benchmarking Standard Group. Available at: http://www.isbg.org

[4] The COSMIC Functional Size Measurement Method Version 3.0: Measurement Manual. September 2007. Available at: http://www.lrgl.uqam.ca/cosmic-ffp

[5] Guidelines for sizing Business Applicatin Software using COSMIC-FFP, version 1.0. December 2005

[6] The COSMIC Functional Size Measurement Method Version 3.0: Method Overview. September 2007. Available at: http://www.lrgl.uqam.ca/cosmic-ffp

[7] COSMIC-FFP Measurement Manual, Version 2.2, January 2003. Available at: http://www.lrgl.uqam.ca/cosmic-ffp

[8] Adel Khelifi, Alain ABran, Luigi Buglione. A System for References for Software Measurements with ISO 19761 (COSMIC-FF). IWSM/MetrKon 2004

[9] Christophe J. Lokan. Function Points. Advances in Computers. M. Zelkwitz (ed), volume 65, chp 7. Academic Press, 2005

[10] Charles Symons. Benefits to the UK software industry of successfully exploiting metrics, and the importance of the COSMIC-FFP method to producing credible metrics. December 2006. Available at www.compaid.com/caiinternet/ezine/symons-cosmic.pdf

[11] van Lamsweerde, Axel, Requirements Engineering - From System Goals to UML Models to Software Specifications", Wiley, January 2009.

[12] Charles Symons. Sizing and estimating for Real-time Software – The COSMIC FFP method. Software Tech September 2006, vol. 9, number 3.

[13] A. Hoshtein, R. Zarnekow, W. B Brenner. ITIL as common practice reference model for IT service management: formal assessment and implications for practice. IEEE International Conference on e-Technology, e-Commerce and e-Service, 2005

[14] International Software Benchmarking Standards Group http://www.isbsg.org

Author CVs

Sanae SAADAOUI

Dr Sanae Saadaoui is a senior researcher at the Centre of Excellence in Information and Communication technologies (CETIC). She is expert in software measurement and process improvement. She is involved in European technology transfer research projects and local industrial services.

Annick Majchrowski

Annick Majchrowski is currently working in the role of researcher/senior project manager at the Centre of Excellence in Information and Communication Technologies in Belgium. As researcher, Annick works on the domain of software and system engineering where she focuses on the quality of software products and software processes. Her research interests are in software process assessments and improvements. As senior project manager, Annick is responsible for service missions in companies with focus on quality management in IT domain.

Christophe Ponsard

Ir. Christophe Ponsard runs the Software and System Engineering at the Centre of Excellence in Information and Communication technologies in Belgium. He coordinates research projects, technology transfer projects and service missions in software process and product quality, model-driven development supporting security and safety and certification. His specific areas of interest and expertise are requirements engineering, rigorous modelling and process measurement.

A Review of Effort Estimation Lineal Mathematical Models for the Software Project Planning Process

Borja Martín[1], Pablo R. Soria[1], María-José Domínguez[1], Marian Fernández[1],
José-Luis Cuadrado[1]
[1] CuBIT, Software Measurement Laboratory,
University of Alcalá, Madrid, Spain
pablo.rsoria@uah.es

Abstract

This technical report presents a review of the software project estimation methods that have been developed across the software engineering history, mainly focused on the effort estimation lineal mathematical models. These methods and models have been classified from new criteria and are based specifically on public dissemination models. For each model is showed its main characteristics, elements and equations that allow us to see as a whole the operation and implementation of each of these effort estimation methods.

Keywords: Public estimation methods, Effort equations, Size equations, time development equations, Lineal mathematical models, Software project planning process.

1 Introduction

Since the first computing developments until the current ones, a fundamental problem has been the fulfillment of certain deadlines for delivery within an established cost, as well as to be able to track and control the projects evolution. Therefore, the establishment of some methods that would enable us to obtain these objectives in a way as realistic and accurate as possible has become an increasingly factor for The Computing Science as a whole, and such methods have been based on knowledge acquired by different disciplines of this science, since Software Engineering to Artificial Engineering.

In addition to produce ever better results in the original objectives, the continuing evolution that the estimation methods have experienced, has also allowed to obtain other benefits such as the upgrading of the projects risk analysis or the possibility of quantitative analysis on the effectiveness of different change proposals of the software construction processes.

Since the 1960´s until now, have been published a large number of models and have also been proposed various classifications of the same based on different criteria. One of the best-known and referenced in the literature is the one proposal by Dr. Boehm in his book Software Engineering Economics [4]. In that book, Boehm proposed the existence of seven estimation methods in such a way that any model could be classified into one of these methods. The sevens methods are listed below:

1. Algorithmic Models

This method brings together the models that provide one or more algorithms that allow us to make the cost estimations of a software product from a set of variables so-called “drivers of cost”. There are five types of algorithms that give rise to five types of models:

a. Lineal. The algorithms are lineal equations.

b. Multiplicative. The algorithms are multiplicative equations.

c. Analytics. The algorithms are a mathematical equation of any kind.

d. Tabular. It was based on the use of a set of tables to determine the value of the input values of the independent variables.

e. Compounds. Incorporates a combination of lineal functions, multiplicative, analytical and tabular.

2. Estimation Based on the Experts Experience

This method involves the consultation to one or more experts to carry out the estimates. Such consultation is done through the implementation of some model of this type.

3. Analogy

This method performed the estimates comparing the current project with the actual data on others of the same features made previously.

4. Parkinson

This method is based on the principle that the cost will be the maximum of available resources.

5. “Price to Win” Method

This method is based on the principle that the cost must be that one able to produce more benefits to the marketing of the product.

6. Estimation Method from Top to Bottom

The estimates are made for the whole project and then subdivide to know the associated with each part of the project.

7. Estimation Method from Bottom to Top

The project is divided in different part and estimations are carried out for each of them. The estimations for the total project are the sum of the parts.

Another well known classification is that of Conte, Dunsmore and Shen [6] comprising four types of models:

1. Historic / Experimental

It refers to those models based on a set of equations proposals by an expert.

2. Statistics

It brings together the models using a regression analysis to establish the relationship between the effort and drivers of cost. It differ two rates depending on the equations used:

a. Lineal. Los The algorithms are lineal equations.

b. Non Lineal. The algorithms are non lineal equations.

3. Theoretic

They are based on theories about how human beings are communicated among themselves, about how the human mind works during the programming process or about mathematical laws that follows the process of developing a software product.

4. Compounds

They incorporate a combination of analytical equations, statistical adjustment data and experts ideas.

The rest of this paper is structured as follows; Section 2 provides a new proposed classification for effort estimation model following the basis lines of previous models. Then Section 3 describes the lineal and public mathematical models that we are going to present and analyze in a chronological order. Finally, conclusions and future research directions are provided in Section 4.

2 New Proposed Classification for Effort Estimation Models

Here we are going to propose a new classification for effort estimation models [7] that although takes some ideas of the proposed previously, presents a new approach under which such methods can be structured into four groups:

1. Mathematical Models

This group would include all those models using mathematical equations to make these estimates. The difference in whether they have been generated by experts is not included because it is assumed that the intervention of experts is essential in the development of the equation. Nor is difference between tabular models, multiplicative, lineal and non lineal. The reasons are from one hand that multiplicative and lineal equations are considered a particular case of the non lineal ones and on the other hand, tabular models or that ones that used lineal equations are already in disuse and all the new published methods use non lineal equations. This method would be similar to the type 1. Algorithmic Models of Boehm or type 4. Compound Models of Conte et al. It was also included within this method type 6. Estimation Methods from Top to Bottom and 7. Estimation Methods from Top to Bottom.

Only the differentiation between Public Models and Owners Models is introduced due to its important consideration, because on the first ones will be possible to make a research work and study, as well as an adequacy to a local environment on the part of the researches or users; and on the second ones will be only possible to know in a general way its functioning or employment method and procedure to acquire or rented. Due to this reason, we will focus our study on the public methods.

2. Estimation based on Experts Experience

The models classified within this method are those that provide a way to extract of the most objective as possible knowledge of experts to carry out the estimations. This is normally done through a set of steps or through the use of questionnaires. It is similar to the type 2. Estimation Based on the Experts Experience by Boehm.

3. Learning Aimed Techniques

This method is based on the use of data collected in previous projects to make these estimates and in this sense is similar to the Mathematical Models, which also used data from previous projects to determine its parameters. But the difference that present models classified in this methodology, a part of not being based on mathematical equations, is that does nor pretend to be of universal application but his goal is to make precise estimates for those projects of the same type of those who have been used to learn. It follows the general idea of type 3. Analogy described by Boehm, but includes the most modern techniques like the Neural Networks.

4. Dynamic Models

These models assume that the cost factors of a software project change over time and make estimates and simulations of such variations. Therefore the basis of this method is very different from the other three, which have a static view of the system.

This paper will focus on the lineal parametric mathematical estimation models particularly on the public disseminated ones, due to its wide use and transparency.

3 Mathematical Models

Mathematical models are based on the development of some mathematical equations that allow, in most universal way as possible, to obtain, from the introduction of a set of an independent variables values such as the size of software product to develop or the users experience; the value of another set of dependent variables, such as the effort or the time needed to develop the product. These equations, which began being lineal in the first models that were developed and that in the latest models are nonlineal, are adjusted through a set of parameters, which, in turn are those that determine the scope of the model.

The development of the parametric mathematical models began in the mid 1960’s and since then, a large amount of models have been developed until now. This model´s evolution has been linked to the progress that has occurred in the age of the computers, with the constant change that has taken place in the hardware and software and especially in the way to perform this last one.

The mathematical models can be classified in public and owners or patented. The first models that were developed used to be public since their equations, parameters, etc., were free dissemination published by the authors, which has allowed its study, analysis and subsequent publications of it, as well as its utilization and calibrated in different environments in which were designed. But as this field took importance, was generating commercial interests, so that among the latest models used in recent years, most of them are owners or patented by companies that operate them. Although these patented models acted as a black box and we can only know their results, most of them tend to be the result of the evolution of some previously free model so it is assumed that their fundamental bases are the mathematical equations in which was based on the original model. In this paper we are going to focus our research only on public models due to its extensively and wide range detail, all this in a chronological order.

3.1 Farr-Zargoski

The official name for this model is Model of Farr and Zargorski. It was first time developed in 1965 by Farr and Zargorsky [8]. This model incorporates 13 factors called Productivity Factors that have importance on the programmer productivity. These factors are determined by a regression analysis of the cost data. The main factors are the number of the dedicated instructions, the type of documents both of internal and external use and the number of words in the database. It is based on the following equation for effort:

$$e = \sum_{i=1}^{13} a_i x_i \qquad (1)$$

Where e represents the Effort measured in MM[1], a_i the value of the parameter *i* corresponding to the variable x_i.

3.2 SDC-Nelson

The official name for this model is SDC which comes from System Development Corporation, the company in which was carried out the study that allowed the development of this model. It is also known as Model of Nelson. It was first time published in 1966 by Nelson [10].

During the development of this model, Nelson identified initially 104 attributes of a software project that had influence on the project cost. Working on a database of 169 real and completed software projects and conducting a statistical analysis of the attributes, he identified the 14 more significant, among which was not the size of the product rather than through the number of subroutines. These 14 drivers of cost are:

1. Lack in the requirement
2. Design stability
3. Percentage of mathematical instructions
4. Percentage of Input – Output instructions
5. Number of subprograms
6. Language used
7. Management Application
8. Unimodular program
9. First program of the computer
10. Concurrent development software
11. Use of random access device
12. Different hardware host
13. Number of personal errors
14. Development for a military organization

This model is based on the following equation for effort:

$$e = \sum_{i=0}^{14} a_i x_i \qquad (2)$$

Where *e* represents the Effort measured in MM, a_i the value of the parameter *i* corresponding to the variable x_i.

3.3 Aron

The official name for this model is Model of Aron. It was first time published in 1969 by Aron [3].

Aron noted that the development path of a software project will be gradually increased until it reaches a maximum, which is located approximately in the test phase, and then decreased until the closing phase of the product. Aron proposes several methods to estimate the costs of the software, these are:

- *Experience*. Similar to the study of cases included in the learning aimed techniques.
- *Work Units.* The total work to do is subdivided into smaller and manageable units for which it is easier to estimate the cost in a more precisely way on the basis of previous experiences with similar units.

[1] MM – Men per Month

- *Quantitative Estimation*. It is the method by which has been classified this model in this paragraph. The total work is divided into small tasks whose implementation is classified in three levels of difficulty: easy, medium and hard, based on the number of interactions with other tasks. Allocating an average productivity to each type of task of 500, 250 and 125 SLOC/MM2 respectively. The effort to each task is calculated by dividing the number of SLOC between appropriate productivity. The total effort is the sum of the effort of the various tasks that make up the project.

This model basis its functioning using the following equation for effort:

$$e = \sum_{i=1}^{n} \frac{si}{pi} \qquad (3)$$

Where e represents the Effort measured in MM, s_i the size in SLOC of the task i, and p_i productivity average of the task in the type to which it belongs.

3.4 Navair

The official name for this model is NAVAIR and comes from NAval AIR Development Center Model. It was first time developed in 1971 by Buck [5].

This model is used to predict the total acquisition cost attending to research, development, test and evaluation of future avionic systems. It uses as effort factors the number of given instructions, the document types, the average experience of the programmers, the number of independent consoles and the percentage of new developed instructions. This model is based using the following equation for effort:

$$e = \sum_{i=1}^{13} a_i x_i \qquad (4)$$

Where e represents the Effort measured in MM, a_i the value of the parameter i corresponding to the variable x_i.

3.5 TRW – Wolverton

The official name for this models is TRW which comes from the company in which was carried out the study that allowed the development of this model. It is also known as Model of Wolverton. It was first time published in 1974 by Wolverton [11].

This model difference six types of software, the development of a source line of code will cost depending of the software type which it belongs:

1. Control
2. Input – Output
3. Pre – Postprocessor
4. Mathematical Calculations
5. Data Management
6. Real Time

In addition to each type, it introduces six difficulty degrees of the follows; if the same team has already

[2] SLOC/MM - Source Lines of Code / Men per Month

developed similar programs, it is considered that the current program is obsolete, otherwise is new. For each of the two types is assigned a In addition to each type, it introduces six difficulty degrees of the follows; if the same team has already developed similar programs, it is considered that the current program is obsolete, otherwise is new. For each of the two types is assigned a

$$c = \sum_{i=1}^{6} s_i c_i \qquad (5)$$

Where *c* represents the cost, s_i the size in SLOC of the modules of the type *i*, and c_i the cost of a line of code of the type *i*.

3.6 ESD

The official name for this model is ESD which comes from Electronic System Division, the company in which the study that allowed the development of this model was made. It was first time published in 1975 by the Air Forces System Command [1].

This model differences six types of software, the development of a line of source code will have a cost according to the type of software to which it belongs. Also, for each type it introduces six difficulty grades:

1. Control
2. Input - Output
3. Pre – Postprocessor
4. Mathematical calculations
5. Data management
6. Real time

If similar programs have already been developed previously for the same team, it is considered that the current program is old, otherwise is new. For each of two types a different scale of values will be allocated. As it is an obsolete program as it is new, a development difficulty of easy, medium or high is allocated to them. It is based on the following equation for effort:

$$c = \sum_{i=1}^{6} s_i c_i \qquad (6)$$

Where *c* represents the cost, s_i the size in SLOC of the modules from type *i,* and c_i the cost of a source line of code of the type *i*.

3.7 SLICE

The official name for this model is SLICE which comes from System LIfe Cycle Estimation. It was first time developed in 1977 by Kustanowitz [9].

This model is based on the development of a project profile, determining the major phases of the development process and assigning a percentage of total effort to each of these phases. Then, the ranges (maximum and minimum) of expected productivity in MM are determined based on the type of application developed, experience, development environment and the programming language used in each of these phases. Finally it is estimated the total number of SLOC that the project will have. With these data are applied calculation equations. The total effort is obtained adding the management costs and documentation. This model is based on the following equations:

Effort:

$$e = \sum_{i=1}^{n} \frac{s_i}{p_i} \qquad (7)$$

Where *e* represents the Effort measured in MM, s_i the size of SLOC for the modules of type *i,* p_i the productivity for the modules of type *i*.

Development Time:

$$t = \sqrt{e} \qquad (8)$$

Where *t* is the development time in months.

3.8 Function Points

The official name for this model is Function Points. It was first time developed in 1979 by Albrech [2].

The Function Points method is developed as an alternative to the estimation of the software product size through SLOC. Function Points have major level of abstraction than the SLOC, attending to aspects such as the number of input transaction types or the number of different reports generated by the system. When they were presented, function points constituted a complete model for effort estimation and the equation that is gathered here corresponds to this model. Currently this estimation method of function points is used to determine the size of the software that is going to be developed, which will be used as an input variable for some other specific model of effort estimation.

Function Points represent some advantages against the SLOC, this way for example we can estimate them earlier in the life cycle since only it is necessary to have the requisites definition document, which is very interesting if function points are used as input in an effort estimation model and development time, since these two data could be known with good approach and very quickly. Another advantage is that they can be calculated by non technical members of the development team. Also, function points avoid the effects of the coding language and other differences in the implementation.

The calculation of Function Points is performed in two phases:

1. Classify the user's functions under its category and calculate the not adjusted function points attending to the level of information processed by each function, which can be simple, medium and complex. For each level and function, attending to its category there will be a natural number corresponding to the assignable function points to this function.

   Currently there are 5 function categories (In the first article only four were defined):
   1. Internal Inputs (IT)
   2. External Outputs (OT)
   3. Internal logical files (FT)
   4. External interfaz files (EI)
   5. External queries (QT)

2. Adjust the function points attending to the application complexity. 14 complexity featured were analyzed:
   1. Data communications (C1)
   2. Distributed functions (C2)
   3. Execution (C3)
   4. Very used configuration (C4)

5. Transaction speed (C5)
6. On-line data Input (C6)
7. End user efficiency (C7)
8. On-line update (C8)
9. Complex processing (C9)
10. Reutilization (C10)
11. Easy Installation (C11)
12. Easy implementation (C12)
13. Multiple localization (C13)
14. Change facility (C14)

Each one with a variation range:

1. Do not present or without influence = 0
2. Insignificant influence = 1
3. Moderate influence = 2
4. Medium influence = 3
5. Significant influence = 4
6. Decisive influence = 5

The adjusted Function Points calculation can oscillate in +/- 35% from the original Function Points calculation, and are based on the following equations:

Size: $$s_f = s_i\,(0.65 + 0.01 \textstyle\sum_{i=1}^{14} c_i) \qquad (9)$$

Where s_f represents the final size in Function Points, once adjusted. s_i represents the initial size in not adjusted Function Points. *p* is a complexity adjustment factor. c_i represents each of the 14 complexity factors.

Effort: $$e = -13.39 + 0.0545s \qquad (10)$$

Where *e* represents the Effort measured in MM and *s* the size in FP (Function Points).

4 Conclusions

In this paper, we have presented a review of the software project estimation methods that have been developed across the software engineering history, mainly focused on the effort estimation lineal mathematical models. With this review, we have tried to show how these estimation models have evolved from its origins and the mathematical bases that have followed for its implementation.

Taking a clear view of how these estimation models have been based, then may come to understand how are developing and improving the current ones and lines we can follow to analyze the next.

This is not attempts to bring a new model or assess, we just simply intend to present an evolution across the software engineering history according to the new classification on Lineal models with public dissemination for effort estimation models that we have contributed. Therefore, as futures lines we will assist in upcoming revisions to the private models which have been developed so far and we can also revise Non-Lineal models that have led to a great progress in the improvement of the processes of effort estimation in the software project planning process.

5 References

1. Air Forces System Command: Government / Industry Software Workshop Summary Notes, Editorial Electronic System Division, (1975).

2. Albrech, A.: Measuring Application Development Productivity, In Proceedings of the IBM Application Development Symposium, GUIDE/SHARE, Monterrey (California), (October, 1979), pp. 83-92.

3. Aron, J.: Estimating Resources for Large Systems, In NATO Conference Report on Software Engineering Techniques, Eds. J.N. Buxton y B.Randel, Rome (Italy), (1969).

4. Boehm, B.: Software Engineering Economics, Editorial Prentice Hall, 1981.

5. Buck, F.: A Cost-by-Function Model for Avionics Computer System, In NADC-SD-7088 Vol. 1, Editorial Naval Air Development Center, 1971.

6. Conte, S. D., Dunsmore, H. E., Shen, V. Y.: Software Engineering Metrics and Models. Benjamin / Cumming Co., Inc. Menlo Park. 1986.

7. Cuadrado-Gallego, J. J.: Métodos de Estimación de Proyectos Software, In UC3M-TR-CS-2000-01 Vol. 1, Ed. Dep.Informática UC3M, (Spain), 2000.

8. Farr, L. y Zagorsky, H.: Quantitative Analysis of Programming Cost Factors: A Progress Report, In ICC Symposium Proceedings Economics of Automatic Data Processing, Ed. A.B. Frielink, Editorial North-Holland, Amsterdam (Holland), (1965).

9. Kustanowitz, A.L.: System LIfe Cycle Estimation (SLICE): A New Approach to Estimating Resources for Application Program Development, In Proceedings of the IEEE COMSAC, (1997).

10. Nelson, E.: Management Handbook for Estimation of Computer Programming Cost. In AD-648-750. Editorial System Development Corporation, 1966.

11. Wolverton, R.: The Cost of Developing Large Scale Software, In IEEE Transactions on Computers, C-23 (6), (1974), pp.615-636.

Measurement and Quantification are Not the Same: *ISO 15939 and ISO 9126*

Alain Abran, Jean-Marc Desharnais and Juan Jose Cuadrado-Gallego

Abstract

Measurement based on international standards for measurement (i.e. metrology) is not the same as judgmental-based quantification of implicit relationships across a mix of entities and attributes without due consideration of admissible mathematical operations on numbers of different scale types. This paper presents first the Measurement Information Model in ISO 15939 and clarifies next what in it refers to the classical metrology field, and what refers to quantitative analysis of relationships. The paper concludes with two examples of the designs of a measure for ISO 9126, one focusing on a single attribute and the second one attempting to quantify a set of relationships across a number of entities and attributes.

1 Introduction

Software practitioners and researchers alike often forget that numbers are not all created equal. For instance, a number derived from the result of a measurement process which meets the metrology requirements is a quantity expressed with a measurement unit. By contrast, a number derived from a mix of mathematical operations without consideration of measurement units and scale types will still be a number, but it could be a meaningless one (For example: some of the Halstead's metrics (Al Qutaish 2005).

Practitioners may feel good about such a potpourri of numbers from models which appear to take into account a large number of factors (i.e. as in many estimation models and quality models). For example: see the Use Case Points (Ouwerkerk 2006). However, feeling good does not add validity to mathematical operations that are inadmissible in measurement.

In practice, various types of quantitative models produce numbers in outputs (i.e. the outcomes of the models) which do not have the same qualities as numbers which meet the requirements of metrology. A quality model will provide a number which typically depends on a specific selection among a (potentially large) number of alternatives, the assignment of a percentage to each contributing alternative, which assignment is based on the opinion of one person (or a group of persons) and comparison of each contributing alternative with distinct threshold values, which themselves are often defined by opinion as well.

In many instances, in these analysis models

- some, if not all, of the numbers used as inputs to these models are obtained by opinion, rather than from precise measurements (with measurement instruments or from the application of detailed measurement procedures);

- these numbers are combined without explicitly describing the admissible mathematical operations and treatment of the corresponding measurement units; and
- the outcomes of such models are indeed numbers, but they do not have metrological properties, and should be handled very cautiously.

Analysis models like these are quantitative models, but they are not measurement models in the metrological sense. Such differences between quantitative analysis and measurement are not generally discussed in the software engineering literature. In this paper, we discuss these differences using in particular the ISO 15939 Measurement Information Model which contains a metrology related perspective as well as an analysis perspective. These concepts are illustrated with two examples, ones which ended up with the design of a measure of a single attribute and another one which ended up with a set of relationships across a number of entities and attributes.

2 ISO 15939 Measurement Information Model & Metrology

The Measurement Information Model from ISO 19539 sets out the various steps necessary for the design an *information product* when a measurable concept has to be designed and used in practice. Figure 1, when read from the bottom up (Right hand-side), shows the following:

1. A specific *measurement method* has to be designed to obtain *a base measure* for a specific attribute;

2. The values of two or more base measures can be used next in a computational formula (by means of a measurement function) to construct a specific *derived measure*;

3. These derived measures are used next in the context of an *analysis model* of relationships to construct an *indicator*;

4. Then, the indicator (i.e. the number from point 3 above) is used for interpretation purposes to build the information product to meet the information needs. This means that the indicator's value is interpreted within the prescribed context as describing, in the language of the measurement user, an information product for his information needs [ISO 15939].

It is to be noted that, in 2 and 3 above, both the derived measures and the indicator inherit the properties of the mathematical operations on which they are built.

- These numbers are meaningful (i.e. have valid properties) when derived from admissible mathematical operations.
- These numbers are meaningless when derived from inadmissible mathematical operations, or when the measurement units and measurement scale types are not considered correctly within the mathematical operations.

The bottom portion of the ISO 15939 Measurement Information Model can be mapped to the metrology concepts in two steps – see Figure 1 (left-hand side):

1. Data Collection: when a measurement method is used to measure an attribute[1], the corresponding output is the base measure of the specific entity being measured – this then corresponds to the data collection of the base measure for each entity being measured;

2. Data Preparation: when a number of the base measures of the data collected are combined through a measurement function (using agreed-upon mathematical formula and related labels), then the combined units are considered as derived measures. This corresponds, then, to data preparation, prior to the analysis phase.

Data collection: base measures

Every base measure must correspond to a single, distinct, software attribute (i.e. a property of an object or concept). So, identifying the attribute of the entity to be measured and quantifying it through its

[1] Of course, the attribute must be well defined; if not, it is pretty challenging to design an adequate measurement method.

measurement method corresponds to the data collection step – see Figure 3.

Data preparation: derived measures

Depending on the Information Needs, some of the base measures already collected for an entity can be assembled according to a measurement function (e.g. a computational formula) defined for each derived measure – see Figure 2 in the Data Preparation step: A derived measured is therefore the product of a set of measurement units properly combined (through a measurement function).

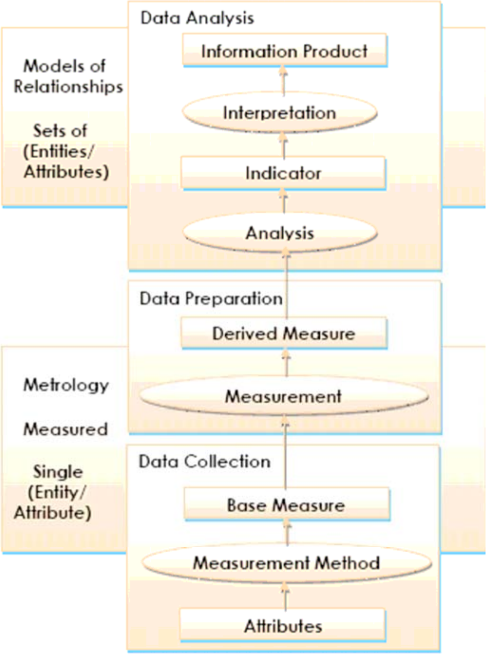


Figure 1: The 2 perspectives of the ISO 15939 Measurement Information Model

If a derived measure is designed bottom-up, the name assigned to this combination of units should correspond to the concept representing the particular combination of measurable attributes. The accuracy of a derived measure (together with the corresponding measurement errors) is directly related to the accuracy of each of its base measures, and how these base measures are mathematically combined. Stated differently, the qualities of the corresponding measuring instrument(s) of the base measures impact the quality of the derived measures. Example: The accuracy of a measurement of velocity will directly depend on the accuracy of its 2 base measures: distance and time.

When their corresponding base measures are not sufficiently well defined, standardized, and instrumented to ensure the accuracy, repeatability, and repetitiveness of measurement results, then, when the same entity (software) is measured by different measurers, the results can potentially be significantly different. It must be noted that a derived measure is descriptive. It does not explain a relationship, nor does it say anything about the strength of such a relationship.

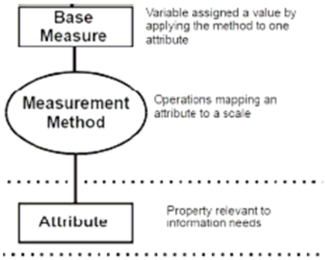

Base Measure
Variable assigned a value by applying the method to one attribute
Measurement Method
Operations mapping an attribute to a scale
Attribute
Property relevant to information needs

Figure 2: The data collection of a base measure for an attribute (ISO 15939)

Example of a Derived Measure: Velocity: The combination distance traveled over a period of time (e.g. km/hour) is associated with the concept of velocity. Such a derived measure (i.e. velocity) is typically measured by a measuring instrument which:

- captures both base measures simultaneously (that is, distance and time, measured in meters and seconds on a car's speedometer, for instance),
- has an integrating feature which divides the base measures to produce a ratio (time/distance) to represent the velocity concept, and
- has a display feature which shows up the measurement results using a standardized display convention:
  for example, converting meters per second into the universally adopted standard for cars, which is 'kilometers per hour'.

Each of the ISO 9126-proposed derived measures (there are over 250 of them) is defined at a fairly high level as a formula composed of base measures. It must be observed that the result of the mathematical operations must also lead to the combination of the measurement units of its corresponding base measures:
Base Measure 1 (B1): Number of detected failures.
Base Measure 2 (B2): Number of performed test cases.
Derived Measure: B1 / B2, with the following measurement units:

$$\frac{B1}{B2} = \frac{\text{Number of detected failures}}{\text{Number of performed test cases}}$$

3 The Quantification of Relationships in ISO 15939

The top portion of the ISO 15939 Measurement Information Model deals with the analysis (through quantification) of relationships across entities and attributes. This analysis part of the ISO 15939 Measurement Information Model includes two activities:

1. Analysis model: refers to the modeling of the relationships across entities and attributes to derive an indicator of the value of such relationships.
2. Interpretation: the indicator would then be interpreted to produce the Information product that would typically be used next in an evaluation or decision making process.

The metrology-related bottom part of the Measurement Information Model is supported by the set of metrology concepts, as described in (Abran 2005). The upper part of the Measurement Information Model is outside the scope of the VIM, since it deals with the use of the measurement results from the lower part of the model.

3.1 Quantitative Elements of the ISO 15939 Analysis Model

While the bottom part of the ISO 15939 Measurement Information Model deals with metrology-related concepts and refers to the VIM terminology, the top part of it is outside of the scope of the VIM: It deals in particular with the use of measurement results in various evaluation or decision making models. This use of measurement results (which have been obtained from a metrology-based approach is represented very succinctly in ISO 15939 using only two activities (Analysis) Mode and Interpretation and one number: Indicator (the rectangle in Figure 1). In practice, however, this use of measurement results typically involves:

- analysis of the relationships across different measurement results with respect to various conditions within a context, and
- assessment against reference contexts for evaluation and/or decision making.

The intricacies and subtleties of the above are not graphically represented in the Measurement Information Model in Figure 1, but are to be found in the textual descriptions of the following three expressions in ISO 15939 where some of the terms not represented in the model are underlined: Indicator, (Analysis) Model and Decision criteria. A number of concepts within these descriptions do not appear in the Measurement Information Model of ISO 15939, such as: Decision criteria, Assumptions, Expected relationships, Estimates or evaluation, Numerical thresholds or targets and Statistical confidence limits, etc.

The ISO 15939 Definitions for the Use of Measurements Results

Indicator: An indicator is a measure providing an estimate or evaluation of specified attributes derived from a model with respect to defined information needs. Indicators are the basis for analysis and decision making. These are what should be presented to measurement users.

(Analysis) Model: An algorithm or calculation combining one or more base and/or derived measures with associated decision criteria. It is based on an understanding of, or assumptions about, the expected relationship between the component measures and/or their behavior over time. Models produce estimates or evaluations relevant to defined information needs. The scale and measurement method affect the choice of analysis techniques or models used to produce indicators.

Decision criteria: Decision criteria are numerical thresholds or targets used to determine the need for action or further investigation, or to describe the level of confidence in a given result. Decision criteria help to interpret the results of measurement. Decision criteria may be calculated or based on a conceptual understanding of expected behavior. Decision criteria may be derived from historical data, plans, and heuristics, or computed as statistical control limits or statistical confidence limits.

NOTE: Some of the terms not represented in the model in Fig. 1 are underlined above in the descriptions.

3.2 Refined representation of the Analysis Model

It was observed in the previous subsection that a number of the concepts mentioned in the 3 descriptions in the side box are not directly modeled in Figure 1. To facilitate an understanding of the relationships across the many concepts embedded within the ISO 15939 Measurement Information Model, the set of key concepts has been extracted from these three descriptions and modeled in Figure 3. The refined representation of the Analysis Model represented in Figure 3 includes two additional major blocks:

A standard reference model (Figure 3, bottom left), which can include, for instance, an accepted model of the relationships across distinct types of objects of interest. When such a reference model exists, this can be: an industry model, an ISO model or a generally accepted statistical technique (and related mathematical model). This standard reference model would include: the set of formal (or informal and assumed) individual relationships, together with the base or derived measures to be considered as evaluation or decision criteria; the algorithm (mathematical or implied) that combines them in an (implied) criterion.

An organizational reference context (Figure 3, upper left), ideally aligned with the standard reference, with a set of selection criteria and values specific to the organization: this organizational reference would contain the reference values necessary for interpretation. This organizational reference context would include A) a set of reference values specified for this context and B) evaluation or decision criteria with either target values, or specific evaluation scales

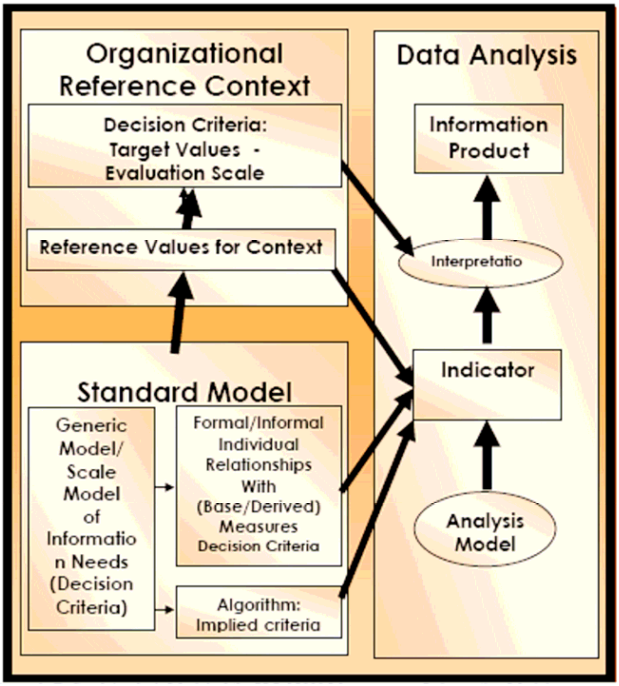

Organizational Reference Context
Decision Criteria: Target Values - Evaluation Scale
Reference Values for Context
Standard Model
Generic Model/ Scale Model of Informatio n Needs (Decision Criteria)
Formal/Informal Individual Relationships With (Base/Derived) Measures Decision Criteria
Algorithm: Implied criteria
Data Analysis
Information Product
Interpretatio
Indicator
Analysis Model

Figure 3: Refined Analysis Model of the ISO 15939 Measurement Information Model

3.3 The (Implicit) Link between Measurement and Quantitative Analysis of Relationships

In the Measurement Information Model of ISO 15939, the link between the two major parts illustrated in Figure 2 (that is, the Metrology-related bottom part of measurement and the Analysis-related upper part of quantification) are not explicitly described. ISO 15939 makes the assumption that this link exists and that it is complete on its own. In practice, the issue might be more complex, in particular in domains where measurement and quantification (or either one) are not yet mature.

In practice, there is no guarantee that what can be measured adequately at the level of base and derived quantities does indeed represent the concepts and relationships that the analysis part of the Measurement Information Model attempts to quantify.

An example of this is the maintainability characteristic in ISO 9126, which is not strictly limited to the software entity itself, but is implicitly related to the entity effort required to maintain such software at a later time.

4 ISO 9126 & ISO 15939

In 1991, the ISO published its first international consensus on the terminology for the quality characteristics for software product evaluation: ISO 9126 – *Software Product Evaluation – Quality Characteristics and Guidelines* for their Use. From 2001 to 2004, the ISO published an expanded four-part version, containing both the ISO quality models and inventories of proposed measures for these models. The current version of the ISO 9126 series consists of one International Standard (ISO 9126 Part 1), and three Technical Reports (ISO TR 9126 Parts 2 to 4). This is complemented by a set of guides in the ISO 14598 series: Developers Guide, Evaluators Guide and Evaluation Modules, etc.

A- The Standard Reference Model
The first document in the ISO 9126 series – *Software Product Quality Model* – contains what can be considered as the **ISO Standard Reference Model** for the quality of software products. This reference model includes three views of the quality of a software product at the highest level – see Figure 4: Internal quality of the software, External quality of the software and Quality in use of the software.

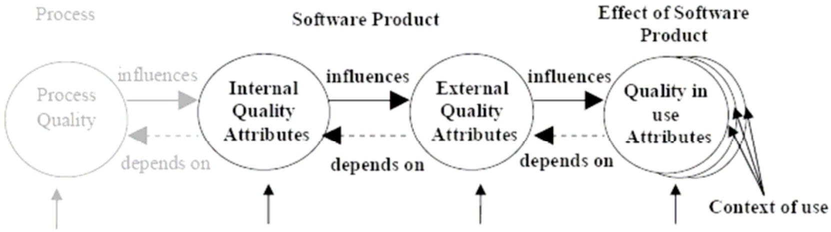


Figure 4: Quality in the software lifecycle – ISO 9126-1

Next, ISO 9126-1 presents two structures of quality models for software product quality:

- a 1st structure for both the internal and external quality models – see Figure 5, and
- a 2nd structure for the quality in use model.

The 1st structure (Figure 5) includes 6 characteristics, subdivided into 27 sub characteristics for internal and external quality [2].

- The 2nd structure includes 4 ‘quality in use’ characteristics: effectiveness, productivity, safety, and satisfaction.
- It must be noticed that ISO 9126 does not provide any reference values for any of its quality characteristics and sub-characteristics.

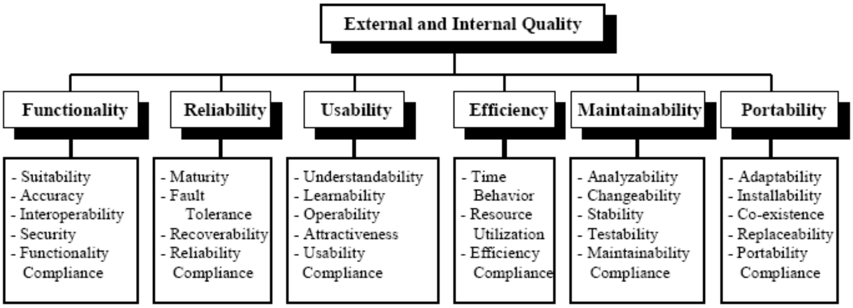


Figure 5: Quality model for External and Internal Quality: characteristics and subcharacteristics – ISO 9126-1

B - An Organizational Reference Context model: Interpretation in ISO 9126 = Evaluation & Decision Making

For the application of the quality model to specific software within an organization, an Organizational Reference Context would typically be set up and used. How to do this is described in the ISO 14598 series, from various perspectives (developers, third party, etc.).

The application of this analysis model (which corresponds to evaluation and decision making on the basis of decision criteria in ISO 9126) is usually performed as a four-step quantification process:

1. Identification of quality-related requirements, that is, the selection of the parts of the ISO quality models that are relevant to a particular context of quality evaluation.

2. Identification of the context of interpretation, that is, the selection of reference values, such values being either generic or specific threshold values, or the determination of targets specified for a particular context.

3. Use of the derived measures from the data preparation phase to fill out the instantiated quality model determined in step 1.

4. Comparison of the results of step 3 with either the set of reference values or the targets determined in step 2 to take a decision based on both the information provided and whatever relevant information is available to the decision maker.

For the set of relationships over the set of objects of interest for the Information Needs, the Analysis Model would typically either:

A- quantify a relationship which is well understood, well described over time, and for which there is a large consensus, or

B- attempt to 'quantify' a relationship (i.e. a concept) for which it is not yet well known how to capture it within a single measurement dimension and a single (base or derived) measure (with its corresponding single measurement unit, or set of such units).

While the ISO 9126 quality models are well described, the relationships across the models, the quality characteristics and sub-characteristics are definitively not well understood and not well described over time. Therefore, any to use in practice any one of such relationships described textually in ISO 9126 represents an 'attempt to quantify' without a prescribed standard or organizational reference context as a well empirically verified foundation. Some of the related issues are described next.

5 The Metrology-related part of ISO 9126

The implementation of an analysis model, including the one from ISO 9126, has to be based on the data collection of base measures (and derived ones, where relevant). The measures available for the data collection for the ISO 9126 quality models are proposed and described in the 3 technical reports in the ISO 9126 series. These reports propose:

- an inventory of +250 derived measures for each quality characteristic or sub characteristic,
- 80 base measures (and corresponding 80 attributes) which are used to build the above +250 derived measures,
- explanations of how to apply and use them, and
- some examples of how to apply them during the software product life cycle.

An example with 2 base measures and a derived measure has been presented in sub section
4.2 of this chapter and is repeated here: Failure density against test cases

Data Collection:
Base Measure 1 (B1): Number of detected failures.
Base Measure 2 (B1): Number of performed test cases.

Data preparation:
Derived Measure: B1 / B2
Name of Derived Measure: Failure density against test cases.

However, as described in the 2003-2005 versions of ISO 9126, most of the attributes to be measured and their corresponding ‘base measures’ are not documented at a detailed enough level to provide sufficient guidance to ensure the accuracy, repeatability, and repetitiveness of measurement results, in the event that the same software is measured by different measurers, which in turn leads to potentially significantly different values. To put it another way, while the numerical assignment rules for each derived measure are described as mathematical operations in the 2003-2005 versions of ISO 9126, neither the base measures for these operations, nor the corresponding quality attributes, have been described with sufficient clarity to ensure the quality of the measurement results.

6 The (missing) links in ISO 9126

This section illustrates the missing links between the quantitative models proposed in ISO 9126 and the detailed measures proposed as the contributors to this quantification. The example selected is that of the **Maturity** sub-characteristic and of the 7 derived measures proposed for this sub-characteristic. The model proposed by ISO to quantify the external quality of a software product contains 6 quality characteristics and 27 sub-characteristics – see Figure 9: One of the 6 quality characteristics is ‘**Reliability’** which is itself composed of 4 sub-characteristics, and one of these 4 sub-characteristics is ‘**Maturity’**.

To quantify this single **Maturity** sub-characteristic, ISO 9126-3 proposes a list of 7 distinct derived measures:

1. Failure density against test cases
2. Estimated latent fault density
3. Fault Density
4. Fault removal
5. Mean time between failures (MTBF)
6. Test coverage
7. Test maturity

A-Metrology: Each of these 7 derived measures is presented only at a fairly abstract level as formulae composed from a set of base measures, themselves lacking detailed descriptions, including the attributes they are attempting to measure. This leaves each of them highly susceptible to individual interpretation: neither the base measures for these operations, nor the corresponding attributes, have been described with sufficient clarity to ensure the quality of the measurement results: they are not documented at a detailed enough level to provide sufficient guidance to ensure the accuracy, repeatability, and repetitiveness of measurement results, in the event that the same software is measured by different measurers, which in turn leads to potentially significantly different values.

Each one of the 7 proposed derived measures are described individually as illustrated in the side box with the 'Failure density against test cases' as an example.

Example: 'Failure density against test cases'

Purpose of this derived measure in ISO 9126: How many failures were detected during defined trial period? Method of application for this derived measure: Count the number of detected failures and performed test cases.

However, none of the embedded base measures are defined precisely in ISO 9126, including failure and test cases.

Analysis model: Neither of ISO 9126 and ISO 14598 (and the upcoming ISO 25000 series) proposes 'standard models of analysis' nor an inventory of 'organizational reference contexts' with reference values and decision criteria: Each organization has to define its own, thereby limiting possibilities of comparisons industry-specific wide or industry-wide.
Therefore, there is no ISO defined context or values for such context. Each group within each organization has to build its own set of values for analysis within a specific context.
The missing link: However, what is the specific contribution of any one of the above 7 derived measures to the Maturity sub-characteristic? Are there some overlaps across the relationship of any of these 7 derived measures, and if so, to what extent? If not all 7 derived measures are mandatory-necessary, which one or which ones are the most representative of the Maturity sub-characteristic, and if so, to which extent? For the **Maturity** sub-characteristic as an example:

- The relationships between the attribute being measured by the derived measure 'failure density against test cases' and the 'maturity' sub-characteristic and the 'reliability' characteristic are not identified, neither described.

- No model of such relationships is proposed either.

Summary: None of the expected links between this (weak) metrology basis for the measurement of the base and derived attributes and the quantification of the quality sub-characteristic (eg. Maturity) and characteristic (e.g. Reliability) is described in ISO 9126.

Hopefully, such links will be described but it will take years of research and development to gain insights into this problem and to develop analysis models based on solid empirical evidence.
In ISO 9126, there are 80 attributes identified as required to be measured as necessary for the + 250 derived measures proposed to quantify the 3 quality models, the 10 corresponding quality characteristics and the 27 quality sub-characteristics. The measurement of one of these attributes, the 'function', is necessary for 38 different derived measures, while another one, the 'user pauses', is needed only in a single derived measure. This section presents now the outcomes of an exercise carried out in a graduate course where it was required to select a attribute from any of the 80 attributes in ISO 9126 and to design a corresponding measurement method.
Using the measurement design methodology presented in (Habra 2008) the graduate students came up with 2 very distinct types of design:

- A design corresponding to the metrology related part of ISO 15939,

- A design which, instead, took the perspective of the analysis of relationships and came up not with the design of a base measure, but with a quantification model of relationships across entities and attributes.

An example of each type of designs is presented next, not because these 2 designs are complete and finalized, but only to illustrate that designers of software measures must beware that measurement and quantification are very distinct concepts, and have different properties.

7 Example 1 of a Metrology Design for a base measure

In ISO 9126 the ‘number of cases’ is necessary in 38 distinct derived measures. To obtain the ‘number of cases’ as a measurement result, it is necessary to have a well defined attribute of what is a ‘case’, and this definition should preferably be the same for each of the 38 distinct derived measure. This design has been described in more details in (Ozcan Top, 2009)

Step 1: Determination of the measurement objectives
The specific objective selected was the design of a measurement method for the size of a ‘case’. The results of the measurement method were intended to be used in the derived measures for the following ISO 9126 characteristics and sub-characteristics – see Table 1.

Character	**Subcharacter Sample**	**Measurable Metric (Intended Usage)**
Reliability	Recoverability	Availability
Functionality	Interoperability	Data exchangeability
Usability	Understandability	Demonstration Accessibility in use
	Learnability	Help frequency
	Operability	Customizability
Maintainability	Analyzability	Status monitoring capability
	Changeability	Parameterized modifiability
	Stability	Change success ratio
	Testability	Availability of built-in test function
Portability	Installability	Ease of installation

Table 1 Related Characteristics and Sub-characteristics which require the measurement of ‘cases’

Step 2: Characterization of the concepts to be measured

The characterization of a concept should initially be based on the findings from a literature review. The literature review started with the definition of ‘cases’ in ISO standards as well as with related definitions in the Usecases literature. Twenty nine relevant references were identified in the literature review, including:

- ISO/IEC FCD 24765 Systems and software engineering – Vocabulary
- ISO/IEC 26514 Systems and software engineering--requirements for designers and developers of user
- ISO/IEC 19761:2003: Software Engineering - COSMIC-FFP: A Functional Size Measurement Method (2003)

Definition and decomposition of the concept
A number of concepts were identified from the literature review. Some of their definitions are presented in the side box. From these, the concept of ‘action’ was identified as the central one from a

measurement perspective.

Definitions of attributes – From the literature review (Ozcan Top, 2009**)**

Case is defined by ISO FCD 24765 Systems and software engineering vocabulary as: "A single-entry, single-exit multiple-way branch that defines a control expression, specifies the processing to be performed for each value of the control expression, and returns control in all instances to the statement immediately following the overall construct" 0.

Use Case is the description of the interaction between an Actor (the initiator of the interaction) and the system itself. It is represented as a sequence of simple steps. Each use case is a complete series of events, described from the point of view of the Actor. Actor, Main scenario, Alternative Paths (Extensions), and Exceptions are the concepts that will be base of the measurement:

An Actor is "someone or something outside the system that either acts on the system – a primary actor – or is acted on by the system – a secondary actor. An actor may be a person, a device, another system or subsystem, or time. Actors represent the different roles that something outside has in its relationship with the system whose functional requirements are being specified."

Preconditions define all the conditions that must be true before the initiation of the use case

Main scenario is the description of the main success scenario in a sequential order. Action is the element of a step that a user performs during a procedure.

Post-conditions "describe what the change in state of the system will be after the use case completes. Postconditions are guaranteed to be true when the use case ends.".

Alternative paths; "Use cases may contain secondary paths or alternative scenarios, which are variations on the main theme. Each tested rule may lead to an alternative path and when there are many rules the permutation of paths increases rapidly.

Sometimes it is better to use conditional logic or activity diagrams to describe use case with many rules and conditions."

Exceptions, is the place "what happens when things go wrong at the system level are described, not using the alternative paths section but in a section of their own." An example of an alternative path would be: "The system recognizes cookie on user's machine", and "Go to step 4 (Main scenario)". An example of an exception path would be: "The system does not recognize user's logon information", and "Go to step 1 (Main path)"

Definition of the Sub Concepts: Three key sub concepts were identified next as follows:

- **Input action**: "Any item, whether internal or external to the project that is required by a process before that process proceeds". "Data received from an external source".

- **Output action**: "Data transmitted to an external destination" 0. "A product, result, or service generated by a process."

- **System action** "set of interrelated or interacting activities which transforms inputs into outputs".

Design of the Meta-Model: Figure 6 presents the meta-model proposed to illustrate the relationships across the concepts and sub-concepts selected to characterize the size of 'cases'.

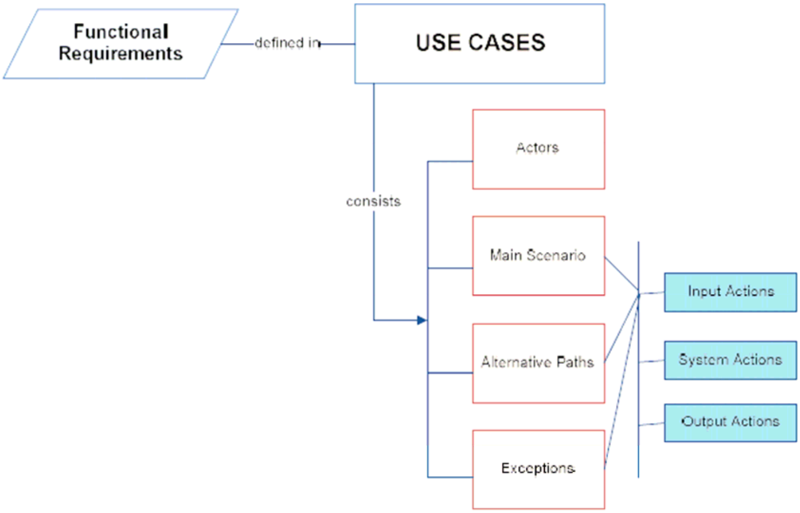


Figure 6: Meta-model of the concepts and sub-concepts for the size of a 'case' (Ozcan Top, 2009)

Numerical Rules: Assignment

The empirical description: The size of a 'case' was defined as the addition of the Input Actions, System Actions and Output Actions. According to this measurement function, each action type (Input Action, System Action, and Output Action) is assigned next a numerical size of 1 Action Unit (A).

Mathematical Expression(s): The above empirical description can now be expressed as a mathematical expression:

Size of a Case = _(Input Actions) + _ (System Actions) + _(Output Actions)

Measurement Scale Type: AU (Action Unit = 1) has a ratio scale measure which means it can be used in statistical analysis, mathematical calculations. These numerical rules are presented in Figure 7.

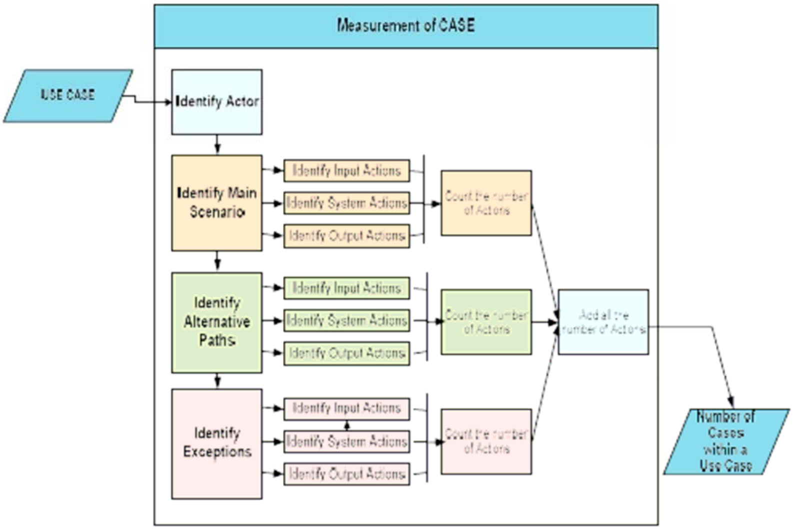


Figure 7: Measurement model of the size of cases within a Usecase (Ozcan Top, 2009)

8- Example 2: Design of an analysis model of relationships across entities and attributes

In ISO 9126, the number of ‘error messages’ is necessary to measure the ‘Efficiency’ and ‘Resource Utilization’.

Step 1: Determination of the measurement objectives
The specific objective selected was the design of a measurement method for the efficiency of ‘error messages’. The results of the measurement method were intended to be used in the derive measures for the following ISO 9126 characteristics and sub-characteristics: Usability, Operability, Understandability and Learnability.

Step 2: Characterization of the concepts to be measured
Definition and Decomposition of the Concepts: A number of concepts were identified from the literature review. Two of the main concepts identified were: Message Appearance and Message Content. In turn, each of these concepts can be decomposed in a number of sub-concepts.
Design of the Meta-Model: The identification of the relationships across the concepts and sub-concepts are illustrated in Figure 8.

Step 3: Assignment of Numerical Rules
Empirical description
The ‘effectiveness’ of an error message was defined as the quantification of both the appearance and the content or error message, on the basis of the quantification of each of their own sub-concepts, as illustrated in the meta-model presented in Figure 8. All of these sub-concepts were themselves quantified individually using their own set of rankings assigned by the person in charge of evaluating the effectiveness of the error message. For some of these sub-concepts the ranking selected were from o to 4, others from 0 to 5 and for some other ones, from 0 to 10.

Mathematical Expression(s)
Effectiveness of an error message is calculated based on measuring the sub concepts:

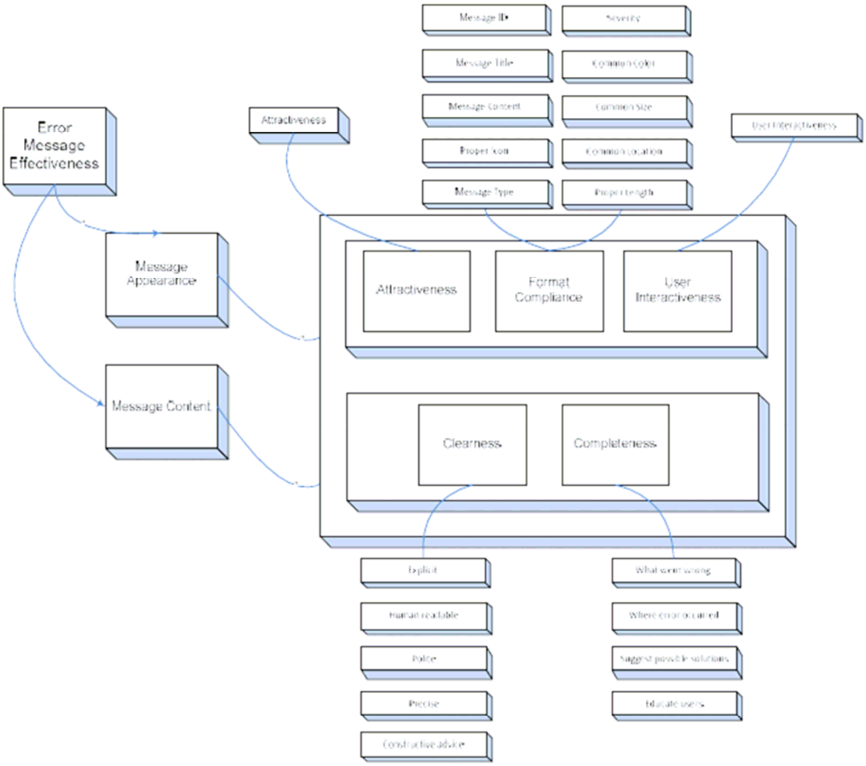


Figure 8: Attributes and Relationships (Dikici 2009)

The sub concepts are to be measured based on the rules specified in Table 2.

- The 4 concepts are each assigned a relative weight (as a percentage)
- The 21 sub concepts (within these 4 concepts) are next assigned a range of rankings, starting at 0, and up to 4, 5 and 10 – see Table 2.

In this specific numerical assignment rule, each sub-concept has an equivalent range within a concept (e.g. from 0-4 for the 10 subconcepts participating to the upper concept which itself was assigned a weight of 40).

Concept	Sub-concept	Weight	Rank ranges
Message Appearance			
Attractiveness		10	
	Attractiveness		0-10
Format Compliance		40	

	Message ID		0/4
	Message Title		0/4
	Message Content		0/4
	Proper Icon		0/4
	Message Type		0/4
	Severity		0/4
	Common Color		0-4
	Common Size		0-4
	Common Location		0-4
	Proper Length		0-4
User Interactiveness		5	
	User Interactiveness		0-5
Message Content			
Clearness		25	
	Explicit		0-5
	Human-readable		0-5
	Polite		0-5
	Precise		0-5
	Constructive advice		0-5
Completeness		20	
	What went wrong		0-5
	Where error occurred		0-5
	Suggest possible solutions		0-5
	Educate users		0-5
		Total	
		Effectiveness %	

Table 2: The numerical assignment structure with weights and ranges of ranking (Dikici 2009)

Measurement Scale Type

The numerical structure above is often used in practice in the evaluation of software quality based on a number of concepts and sub-concepts. However, being used often in practice is no guarantee that this is the most appropriate mathematical structure. In particular, the scale type of the end results of this set of numerical assignment rules is challenging to determine without ambiguity:

- The intervals are in increments of 1 (from 0 to 4, for example), but there is no explicit definition of what is an interval of 1, and no explicit and rigorous definitions that subsequent intervals from 2 to 4 are indeed equal intervals.

- Next, in practice, the selection of an interval is typically judgmentally based, and such a selection would often vary across people selecting a specific value, and may even vary if he same person was to select again a value let us say a week later.

In practice the corresponding values can certainly be considered as ordering values, but considering them as being on a ratio scale type would be somewhat far stretched.

Finally, it must be observed that each of the 21 sub-concepts is different, and if measured adequately, they would each have their distinct measurement units and measurement scales: you would not then be able to add them up (since they do not have the same measurement units).

Therefore, adding up the values assigned to anyone of the 21 sub-concepts does not correspond to a measurement exercise. Adding them up is a quantification, but without the rigor and meaningfulness of measurement with the rigor of metrology. This is typical of many quantification exercises whereas weights and 'points' are somewhat arbitrarily assigned – see (Ouwerkerk 2006; Abran 1994, 1996).

8 Summary

Practitioners and managers can have much more confidence on measurements results based on international standards for measurement (i.e. metrology), than on judgmental-based quantification of implicit relationships across a mix of entities and attributes without due consideration of admissible mathematical operations on numbers of different scale types.

Measurement based on international standards for measurement (i.e. metrology) is not the same as judgmental-based quantification of implicit relationships across a mix of entities and attributes without due consideration of admissible mathematical operations on numbers of different scale types. This paper has presented first the Measurement Information Model in ISO 15939 and has clarified what in it refers to the classical metrology field, and what refers to quantitative analysis of relationships. The paper has presented next two examples of the designs of a measure for ISO 9126.

The first design focused at design a measure for a single attribute by looking at what was similar across the entities to be measured, thereby ending up with a single concept which was simple to add up for measurement purpose, without getting into trouble with distinct measurement units and scale types. Such a rigor is not typical in software measurement, to the exception of the design of the COSMIC – ISO 19761 (ISO 19761).

By contrast, the second design attempted to quantify a set of relationships across a number of entities and attributes, in an approach often found in practice in software engineering: however, doing so provide numbers which typically depend on a specific selection among a (potentially large) number of alternatives, the assignment of a percentage to each contributing alternative, which assignment is based on the opinion of one person (or a group of persons) and comparison of each contributing alternative with distinct threshold values, which themselves are often defined by opinion as well. Such numbers, when combined, do not necessarily consider adequately the admissible mathematical operations and treatment of the corresponding measurement units: at times, the outcomes of such models are meaningless numbers.

References

Abran, A., Al-Qutaish, RE., Desharnais, J.M., Habra, N., An Information Model for Software Quality Measurement with ISO Standards, International Conference on Software Development - SWEDC-REK, Reykjavik, Iceland , University of Iceland, 2005, pp. 104-116.

Abran, A., Robillard, P.N., Function Points Analysis: An Empirical Study of its Measurement Processes, IEEE Transactions on Software Engineering, vol. 22, 1996, pp. 895-909.

A. Abran, P. Robillard, “Function Points: a Study of Their Measurement Processes and Scale Transformations”, Journal of Systems and Software, vol. 25, issue 2, p. 171, 1994.

Abran, A., Sellami, A., Suryn, W., "Metrology, Measurement and Metrics in Software Engineering", 9th International Software Metrics Symposium - METRICS'03, IEEE Computer Press, Sydney (Australia), September 2003, pp. 2-11.

Al Qutaish, R., Abran, A., “An Analysis of the Designs and Definitions of Halstead Metrics”, 15th International Workshop on Software Measurement – IWSM 2005, Montréal (Canada), Shaker Verlag, Sept. 12-14, 2005, pp. 337-352.

Carnahan, L., Carver, G., Gray, M., Hogan, M., Hopp, T., Horlick, J., Lyon, G., and Messina, E., "Metrology for Information Technology", StandardView, vol. 5, No. 3, 1997, pp. 103-109.

Dikici, P., 'Error Message Effectiveness Measurement', Term session paper, Graduate course 'SM517 Software Measures', professor JM Desharnais, METU Informatics Institute, Ankara, Turkey, 2009.

Fenton, N., Pfleeger, SL, Software Metrics: A Rigorous and Practical Approach, 2nd edition, PWS Publishing Co., London, 1997, pp. 638.

Gray, M. M., "Applicability of Metrology to Information Technology," Journal of Research of the National Institute of Standards and Technology, vol. 104, No. 6, 1999, pp. 567-578.

Habra, N., Abran, A., Lopez, M., Sellami, A., A Framework for the Design and Verification of Software Measurement Methods, Journal of Systems and Software, Elsevier, vol. 81, issue no.5, May 2008, pp. 633-648.

ISO/IEC 9126-1: 2001. Software Engineering - Product Quality - Part 1: Quality Model, International Organization for Standardization – ISO, Geneva, 2001.

ISO/IEC TR 9126-2: 2003. Software Engineering - Product Quality - Part 2: External Metrics, International Organization for Standardization – ISO, Geneva, 2003.

ISO/IEC TR 9126-3: 2003. Software Engineering - Product Quality - Part 3: Internal Metrics, International Organization for Standardization – ISO, Geneva, 2003.

ISO/IEC TR 9126-4: 2004. Software Engineering - Product Quality - Part 4: Quality in Use Metric, International Organization for Standardization – ISO, Geneva, 2004.

ISO/IEC 15939: 2002, Information Technology - Software Engineering - Software Measurement Process, International Organization for Standardization, Geneva, 2002.

Ouwerkerk, J., Abran, A., ‘Evaluation of the Design of Use Case Points (UCP)’, MENSURA2006, Conference Proceedings edited by the Publish Service of the University of Cádiz www.uca.es/publicaciones Nov. 4-5, 2006, Cadiz (Spain), pp. 83-97.

O. Ozcan Top, 'Measurement Method for CASE', Term session paper, Graduate course 'SM517 Software Measures', professor JM Desharnais, METU Informatics Institute, Ankara, Turkey, 2009.

Towards Agile Product Derivation in Software Product Line Engineering

Padraig O'Leary[1], Fergal McCaffery[2], Ita Richardson[1], Steffen Thiel[1]
[1]Lero, the Irish Software Engineering Research Centre, University of Limerick, Ireland
[2]Dundalk Institute of Technology, Dundalk, Ireland

{padraig.oleary, ita.richardson, steffen.thiel}@lero.ie
fergal.mccaffery@dkit.ie

Abstract. Software Product Lines (SPL) and Agile practices have emerged as new paradigms for developing software. Both approaches share common goals; such as improving productivity, reducing time to market, decreasing development costs and increasing customer satisfaction. These common goals provide the motivation for this research. We believe that integrating Agile practices into SPL can bring a balance between agility and formalism. However, there has been little research on such integration. We have been researching the potential of integrating Agile approaches in one of the key SPL process areas, product derivation. In this paper we present an outline of our Agile framework for product derivation that was developed through industry based case study research.

Keywords: Software Product Lines, Product Derivation, Agile Approaches

1 Introduction

Both Agile and Software Product Lines (SPL) development paradigms are being promoted as means of reducing time to market, increasing productivity, and gaining cost effectiveness and efficiency of software development efforts [1]. Furthermore, both approaches assume that requirement changes will occur and can be managed effectively [1]. These goals (shared by Agile and SPL) open the possibilities of introducing Agile practices into SPL activities. There are, however, several challenges involved in integrating Agile approaches in SPL development due to certain differences that exist in the philosophies of both approaches such as design and change management strategies [1, 2]. Moreover, Agile approaches do not purpose to develop flexible artefacts for reuse [2, 3] or develop and maintain rigorous and extensive documentation as required by SPL [3].

Our research in SPL is aimed at improving the Product Derivation (PD) process, which purports to develop new products by utilizing core assets of a SPL such as feature models, architecture models, and code artefacts [4], through the adoption of Agile practices.

In this paper we present our research results on the development of an Agile Framework for Product Derivation (AFPD). We decided to concentrate on product derivation as it is considered one of the most important and challenging SPL "activities" [5], and the activity which has the most to gain from the successful implementation of agile practices. We believe that any successful effort to introduce Agile practices in the product derivation process can make SPL significantly more effective and efficient. While some research in the area of Agile SPL has been reported [1-3, 6-8], there has been little research conducted on the use of Agile approaches in the product derivation process.

The remainder of this paper is organised as follows: Section 2 describes the key concepts of SPL and Agile practices. Section 3 discusses the research methodology. Section 4 presents an overview of our Agile Product Derivation Framework. In Section 5, we discuss in detail the Agile aspects of the AFPD. The paper concludes in Section 6 with a summary and an outlook of future work.

2 Background and Motivation

In the following section, we discuss the main concepts of Agile and SPL that underpins our proposal for integrating the two.

2.1 Software Product Lines

A 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 that are developed from a common set of core assets in a prescribed way [6]. The SPL approach makes a distinction between domain engineering, where a common platform for an arbitrary number of products is designed and implemented, and application engineering, where a product is derived based on the platform components [8]. It is during application engineering that the individual products within a product line are constructed. The process of creating these individual products using the shared artefacts is known as the product derivation process [4].

The underlying assumption of product derivation is that "the investments required for building the reusable assets during domain engineering are outweighed by the benefits of rapid derivation of individual products" [4]. This assumption might not hold if inefficient derivation practices diminish the expected gains.

A number of publications discuss the difficulties associated with product derivation. Hotz *et al.* [9] describe the process as "slow and error prone even if no new development is involved". Deelstra *et al.* [4] observe that the derivation of individual products from shared software assets is still a time-consuming and expensive activity in many organisations. The authors state that "there is a lack of methodological support for application engineering and, consequently, organizations fail to exploit the full benefits of software product families." "Guidance and support are needed to increase efficiency and to deal with the complexity of product derivation" [10].

2.2 Agile Practices

Agile practices have recently gained popularity among large numbers of companies as a mechanism for reducing costs and increasing ability to handle change in dynamic market conditions. Researchers and practitioners have proposed several software development approaches based on the principles of the Agile manifesto [11, 12]. Two of these approaches are: eXtreme Programming (XP) [13] and Scrum [14].

XP evolved from the problems caused by the long development cycles of traditional development models [15]. The individual practices of XP are not new, however, the practices have been collected and lined up to function with each other in a novel way. The term 'extreme' comes from taking these commonsense principles and practices to extreme levels [16].

Scrum provides a project management framework that focuses development into 30-day Sprint cycles in which a specified set of Backlog features are delivered [14]. The core practice in Scrum is the use of daily 15-minute team meetings for coordination and integration. Scrum does not define any specific software development techniques. Scrum concentrates on how team members should function in order to produce good quality code and maintain flexibility in a changing environment.

Although XP and Scrum are based on a common guideline defined by the Agile manifesto, they vary in focus and presentation. XP emphasises technical elements of the development lifecycle, while Scrum concentrates on the project management.

3 Research Approach

The preparatory stage of this research was conducted as an extensive literature review. The research aimed to identify the fundamental practices of product derivation and Agile approaches. The initial results were further developed and assessed through a series of iterative workshops over a four month period. Evidence and feedback from SPL and Agile experts was collected from these organised workshops.

We conducted case study research with Robert Bosch GmbH [1]. We collected data on the product derivation practices of their automotive systems. The systems produced consisted of both hardware (such as processors, sensors, connectors, and housing) and software. Many of the requirements were derived from market segments, such as low cost or high cost customers or from regulatory requirements.

Based on knowledge garnered on the derivation practices within the company, we identified areas with potential for the integration of Agile methods. The output of this research was a technical report [17] where we documented our recommendations on the use of Agile practices within Bosch automotive business units.

The research was further developed through two research collaborations. The first was a six month visit to LASSY[2]; where AFPD and FIDJI [18] were mapped. The second was a collaboration project with Doppler Laboratory where we investigated the application of their DOPLER^{UCon} [10] tool within the AFPD

4 Agile Framework for Product Derivation

Product derivation approaches in the literature [4, 19-21] and industry practice observed through this research (c.f. Section 3), typically follow a phased structure. These phases are broadly speaking requirements analysis, product configuration and artefact reuse, and finally product specific development and testing. These phases are reflected in the structure of the AFPD. Through our research into Agile methods we have applied iterative and incremental approaches within this phased lifecycle.

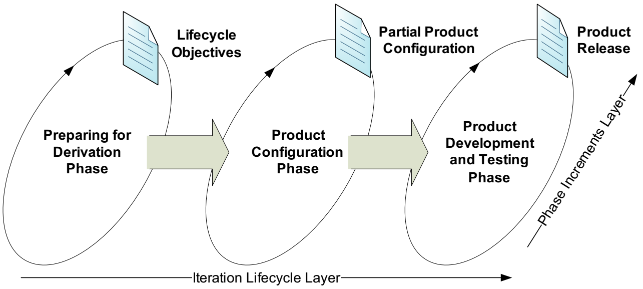

Lifecycle Objectives
Partial Product Configuration
Product Release
Preparing for Derivation Phase
Product Configuration Phase
Product Development and Testing Phase
Phase Increments Layer
Iteration Lifecycle Layer

Fig. 1. Agile Framework for Product Derivation

The three principal phases, consisting of essential activities required during any product derivation project, within the AFPD are: Preparing for Derivation, Product Configuration and Product Development and Testing. Figure 1 provides an overview of these phases, including the main milestone for each phase.

Preparing for Derivation Phase determines the objectives and manages the project. The phase forms the product-specific requirements based on customer requirements and negotiation with the platform team. Requirements are prioritized and assigned to development iterations.

[1] http://www.bosch.com

[2] Laboratory of Advanced Software Systems (LASSY), University of Luxembourg

Product Configuration purports to create a partial product configuration based on the product-specific requirements and by using the available core assets. The aim of this phase is to maximize reuse of the platform assets.

During *Product Development and Testing*, product specific development is undertaken. The product is tested to ensure it satisfies customer expectations.

There two major layers to the AFPD (c.f. Figure 1). These are the phase increments layer and iteration lifecycle layer. Phase increments are short units of work on a particular aspect of the derivation process i.e. configuring platform components. The iterative lifecycle layer structures these phase increments to deliver stable builds of the product that incrementally progress towards the iteration objectives. These iterations result in regular product releases.

The next section discusses expands on the Agile aspects of the AFPD.

5 Increasing Agile in Product Derivation

In this section, we discuss the following Agile elements of the AFPD:

- Adoption of Early and Continuous Delivery Strategy;
- Automation of Product Derivation;
- Product Derivation Iterations;
- Agile Testing Techniques.

We describe how these elements were identified and the benefits that they can bring to product derivation.

5.1 Adoption of Early and Continuous Delivery Strategy

Typically, implementing product specific features can be time consuming. Firstly, product construction can be substantially delayed due to the Change Control Board (CCB). The CCB scopes new development to gauge the reusability of a requested feature within the product line. Secondly, development is further delayed if the Product Team defers implementing a feature until the platform team implement the requested platform changes at the product level.

In the AFPD we adopt the Agile principle of “early and continuous delivery of valuable software”. The product team implement changes at product level. The Platform Team subsequently mine any changes from the product if there is reuse potential.

In Bosch we observed this Agile principle in action. To facilitate early and continuous delivery of software, the product team would not wait for scoping decisions from the CCB. Rather, the product team would negotiate a new platform interface containing required extensions to facilitate new product components before proceeding to develop in parallel against the platform team. When the platform extensions had been implemented and the new platform was released, the product team would check for compatibility issues with newly developed components.

We recommended [17] the adoption of the Agile practice of pair programming for customer specific components. Pair programming is suitable for implementing and reviewing any changes at the product level [6]. This helps to produce better quality product code and consequently, improved code for any features that are mined for the platform.

5.2 Automation of Product Derivation

Automated support for product derivation is a necessity for managing the complexity and variability inherent in software product lines and according to Kurmann [6], automation is the most important aspect of an Agile software product line. Automated development approaches facilitate the Agile Principle “*Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.*” [12], as automated development techniques allow product teams to implement changing customer requirements late in the development lifecycle and automation enables these changes to be implemented quickly.

However current process models and tools for automation do not integrate well. All the stakeholders involved in product derivation are supported in their tasks by different approaches and different automation tools. Because of the difficulty of integrating these different approaches and tools, product derivation can quickly become an error-prone and tedious task.

In our research collaboration with Dopler Laboratory (c.f. Section 3) we investigated how DOPLER^{UCon} [10] tool could be used within the AFPD. We were particularly interested in its ability to facilitate Agile approaches. For instance, we observed that while the DOPLER^{UCon} tool does not directly support *iterative development* cycles by defining additional attributes for requirements it could be used to allocate specific requirements to specific iterations.

5.3 Product Derivation Iterations

The identification of product derivation iterations is a key aspect of deriving high quality, customer satisfying products. According to Carbon et al. [2] when adopting a SPL approach, an organisation is capable of producing a first version of a product for a specific customer, including the core functionality, quicker than other software development methods. Because of the approved quality of the reusable assets, the customer can get a high quality product that can be used and evaluated to give feedback. In further iterations, new functionality can be added to the scope of the product line or product specific features can be implemented [2].

In a technical report to Bosch [17], we recommended that they could benefit from applying the planning game practice from the XP methodology for the management of their product iterations during the *Preparing for Derivation* phase. This would assist them in gathering and negotiating product specific requirements. During customer negotiation requirements are prioritised and allocated to specific iterations based on priority.

5.4 Agile Testing Techniques

Agile methods propose that testing is carried out frequently, as this helps Agile developers keep their code as error free as possible. We have adopted a phased testing approach in the AFPD. Based on the principles of integration testing suggested by Muccini [22], the structure and nature of the elements in a product line are leveraged. Firstly, integrate the partial configuration and use a traditional approach to integration testing. Then, based on the observation that at least the partial product configuration works properly, we can incorporate the other product elements. Product construction continues in a phased assembly test approach. For systems testing of partial or fully assembled products traditional system testing techniques can be utilized as no SPL specific methods exist.

6 Conclusion and Future Work

Our research is motivated by the fact that despite the widespread adoption of SPL within industry, product derivation remains an expensive and error-prone activity [23, 24]. We believe that the adoption of Agile practices can improve the product derivation process. The Agile Framework for Product Derivation provides a means of supporting this adoption.

The development of the framework is a response to calls from industry for research into this area [25]. The integrated Agile framework could solve many of the problems associated with product derivation's complex and cumbersome nature.

The framework is a lightweight approach to product derivation, minimising the amount of up-front investment required making SPL more accessible to small organisations with limited resources. The framework may benefit larger organisations by bringing a balance between formalism and agility, helping individual product teams deliver products with the best possible quality. A combination of Agile and SPL is expected to create a leaner but more disciplined product derivation process [6].

Our future work includes an ongoing investigation into the benefits of combining Agile and SPL approaches and the validation of our framework, particularly with respect to the expected return on investment.

7 Acknowledgements

This work is partially funded by IRCSET under grant no. RS/06/167 and by Science Foundation Ireland under grant no.s 03/CE2/I303_1 and 07/SK/I1299.

References

1. Tian, K., Cooper, K.: Agile and Software Product Line Methods: Are They So Different? : 1st International Workshop on Agile Product Line Engineering, Maryland, USA (2006)
2. Carbon, R., Lindvall, M., Muthig, D., Costa, P.: Integrating Product Line Engineering and Agile Methods: Flexible Design Up-front vs. Incremental Design. 1st International Workshop on Agile Product Line Engineering (APLE'06), Maryland, USA (2006)
3. Navarrete, F., Botella, P., Xavier, F.: An Approach to Reconcile the Agile and CMMI Context in Product Line Development. 1st International Workshop on Agile Product Line Engineering (APLE'06) Maryland, USA (2006)
4. Deelstra, S., Sinnema, M., Bosch, J.: Product Derivation in Software Product Families: A Case Study. J. Syst. Softw., Vol. 74. Elsevier Science Inc., New York, NY, USA (2005) 173-194
5. Griss, M.L.: Implementing Product-Line Features with Component Reuse. Springer-Verlag, London, UK (2000)
6. Kurmann, R.: Agile SPL-SCM Agile Software Product Line Configuration and Release Management. 1st International Workshop on Agile Product Line Engineering (APLE'06). Maryland, USA (2006)
7. Noor, M., Rabiser, R., Grünbacher, P.: A Collaborative Approach for Reengineering-based Product Line Scoping. 1st International Workshop on Agile Product Line Engineering (APLE'06), Maryland, USA (2006)
8. Trinidad, P., Benavides, D., Ruiz-Cortes, A., Segura, S.: Explanations for Agile Feature Models. 1st International Workshop on Agile Product Line Engineering (APLE'06), Maryland, USA (2006)
9. Hotz, L., Gunter, A., Krebs, T.: A Knowledge-based Product Derivation Process and some Ideas how to Integrate Product Development. Proc. of Software Variability Management Workshop, Groningen, The Netherlands (2003)
10. Rabiser, R., Grünbacher, P., Dhungana, D.: Supporting Product Derivation by Adapting and Augmenting Variability Models. 11th International SPLC, Kyoto, Japan (2007)
11. Abrahamsson, P., Warsta, J., Siponen, M.T., Ronkainen, J.: New Directions on Agile Methods: A Comparative Analysis. ICSE 2003, Portland, USA (2003)
12. Manifesto for Agile Software Development. http://agilemanifesto.org/
13. Beck, K., Andres, C.: Extreme Programming Explained : Embrace Change (2nd Edition). Addison-Wesley Professional (2004)
14. Schwaber, K., Beedle, M.: Agile Software Development with Scrum. Prentice Hall PTR (2001)
15. Beck, K.: Embracing Change with Extreme Programming. IEEE Computer Society Press, Vol. 32 (1999) 70 - 77
16. Beck, K.: Extreme Programming Explained: Embrace Change. Addison-Wesley Longman Publishing Co., Inc, Boston, MA, USA (1999)
17. O'Leary, P., Thiel, S., Botterweck, G., Richardson, I.: Bosch AB10 Technical Report. LERO The Irish Software Engineering Research Centre (2008) 26
18. Perrouin, G., Klein, J., Guelfi, N., Jezequel, J.M.: Reconciling Automation and Flexibility in Product Derivation. Software Product Line Conference, 2008. SPLC '08. 12th International (2008) 339-348
19. Bayer, J., Gacek, C., Muthig, D., Widen, T.: PuLSE-I: Deriving instances from a product line infrastructure. Engineering of Computer Based Systems, 2000. (ECBS 2000) Proceedings. Seventh IEEE International Conference and Workshopon the (2000) 237-245
20. Czarnecki, K., Helson, S., Eisenecker, U.W.: Staged configuration using feature models. Proc. of the 3rd International Software Product Line Conference (SPLC 2004). Springer Berlin Heidelberg, Boston, MA, USA (2004) 266-283
21. Halmans, G., Pohl, K.: Communicating the Variability of a Software-Product Family to Customers. Informatik - Forschung und Entwicklung **18** (2003) 113-131
22. Muccini, H., van der Hoek, A.: Towards Testing Product Line Architectures. ETAPS2003. In Electronic Notes of Theoretical Computer Science, Warsaw, Poland (2003)
23. Deelstra, S., Sinnema, M., Bosch, J.: Experiences in Software Product Families: Problems and Issues During Product Derivation. Software Product Lines, Third International Conference. Springer, Boston, MA, USA (2004)
24. Hotz, L., Wolter, K., Krebs, T.: Configuration in Industrial Product Families: The ConIPF Methodology. IOS Press, Inc. (2006)
25. BOSCH: Presentation: Product Derivation - Research Interests within BOSCH. In: LERO (ed.): (2006)

Being Agile when Developing Software Components and Component-Based Systems – Experiences from Industry

Iva Krasteva, Rikard Land, A. S. M. Sajeev

Abstract

Building software from components has some potential benefits in terms of reusing proven solutions and reducing time-to-market. However, component-based paradigm is sometimes in conflict with certain agile principles and practices, such as being responsive to change. This paper presents the results of a web-based survey investigating how industry today tries to manage these challenges. Based on the same data we also describe the preferences of industry professionals for improving development processes in their projects by introducing certain practices. A quantitative analysis using paired-samples t-tests has shown significant differences between current practices and process preferences for both component developers and component users, and the conclusion is that practitioners in general want less rigidity than in their respective currently performed practices.

Keywords

Agile practices, agile principles, component-based software

1 Introduction

Agile software development and component-based development are two paradigms that have gained much attention in the recent years. They both promise to increase system quality, development speed and flexibility, but so far little has been published on the combination of the two approaches.

This paper shows how industry today copes with the challenges of applying agile principles and practices in the development of components and component-based systems [1]. We have previously performed a theoretical study, which investigated potential contradictions between the agile principles and the fundaments of the COTS[1]-based paradigm, in order to provide advice on how agile practices best should be applied to development of COTS components and development of COTS-based software systems, and to identify any challenges [2] and [3]. The paper presents an empirical study of the same issues, and also extends the scope from COTS to software components in general, including in-house developed components for e.g. product lines, and subcontracting agreements. The present paper is based on a web-based survey investigating current processes and practices used in industry. The contribution of the paper is to publish the collected experiences of organizations and individuals regarding how to perform agile development of components and component-based systems. The paper conducts the analysis by interpreting the data qualitatively, as well as by using statistical methods.

Related work is presented in the next section. In Section 3 we describe the research method. In Section 4 and Section 5 we discuss how challenges for adopting agile ideas into the development processes of components and component-based systems are addressed. The preferences of practitioners for improving development processes in their projects by introducing certain practices are presented in Section 6. We conclude and outline future research in Section 7.

2 Related Work

Straightforward adoption of agile methodologies for small less-critical systems [4] has proved very effective in the current dynamic business environment. More and more, the agile approach broadens its areas of application to domains and projects previously considered unsuitable for agile development. An EU project intended to scaling up agile approaches in a globally distributed environment is ongoing FLEXI[2], research papers and case studies report on using agile ideas in safety-critical systems and hardware intensive projects EUREKA-ITEA AGILE Project[3]; [5], [6] and [7]. Software Engineering Institute (SEI) published an official report [8] to suggest consolidation of lightweight agile development with its CMMI reference standard.

The research on introducing agile principles and practices in component-based systems and component development itself is still quite sporadic and isolated and most of the suggested solutions are partial. The Evolutionary Process for Integrating COTS-based systems (EPIC) [9] is a framework for building, fielding and supporting COTS-based solutions. It presents an alternative approach for acquisition, management, and engineering practices of COTS-based systems which is based on Rational Unified Process (RUP). Another study [10], which sets EPIC as a ground, extends the research question further towards agile ideas and presents a set of questions that need to be considered to introduce agility into the process from a requirements engineering perspective. Another part of COTS development process, the selection of COTS components and the applicability of agile principles for component selection, is examined by [11]. A description of using CLARiFi system (CLear And Reliable Information for Integration) as an agile approach for retrieving components from large repositories is provided by [12].

[1] COTS = Commercial Off-The-Shelf component

[2] http://flexi-itea2.org/index.html

[3] http://www.agile-itea.org/

3 Research Method

Our underlying assumption is that agile principles and practices are applicable to the development of components and component-based systems, but specific customization is needed to support agile ideas and to respond to particular project characteristics. To study how agile principles and practices can be applied to the development of components and component-based systems, we constructed a web-based questionnaire. Invitation emails were sent to companies that were part of our joint research projects such as FLEXI and NESSI[4] , among others. We received 93 responses of which a total of 33 respondents addressed questions on component-based development to such an extent that the results were usable for this research question, but since the respondents are anonymous we cannot know how many organizations these represent. Because we sent the invitation to participate to some email lists, and encouraged every recipient to further spread the invitation we can neither know the response frequency, nor exactly which organizations are represented. This type of convenience sampling is suitable to collect empirical data exploring "how" question such as ours, but during all our statistical treatment of the data we must bear in mind the limitations imposed by convenience sampling to the generalization of the results. We should also consider the reasons for potential respondents to not respond to the survey. We believe the most important factors are their lack of time and their not having any incitement to respond. More information about the questionnaire, as well as all data, is available as a technical report [13].

The responses are divided into two groups: Component Developers and Component Users where the latter group use or integrate components in the development of systems. There is some overlap between the two groups in the sense that in some projects components are both developed and integrated. From the total of 33 respondents, we have received 13 answers relevant to component development and 24 for component usage. These should in general be regarded as experienced enough to provide accurate and insightful responses, and we can also observer that various types of development processes are used: Of component developers, three respondents are less than 30 years old, eight have more than nine years of experience in IT, and six are involved in adaptive development processes and four in agile. Of the respondents building systems out of components, 13 are between 30 and 40 years old, 8 are above 40, 20 have more that five years of experience in IT, and one fourth each use either adaptive, agile or waterfall methods.

We analyze the results of the survey both qualitatively and quantitatively. The qualitative approach is applied to verify the most important findings of our theoretical reasoning about introducing agile principles and practices to component development and development of component-based systems. The quantitative approach is used to analyse how development processes can be improved by adopting some agile practices. This is done by studying the differences between current practices used by CBSE[5] community and preferences of practitioners towards using agile practices in their projects.

We have designed our study to collect the data which enables us to characterize each response as component development (of COTS components, as a subcontractor, for a product line, etc.), or system development (by assembling COTS, in-house components, through subcontractor agreements, etc.). To support the theoretical findings we gather data in two main directions: contradictions between the approaches and suggestions for application. We used a seven-point Likert scale (with responses varying from 'Very strongly agree' to 'Very strongly disagree') to indicate how strongly respondents agreed or disagreed with statements about current development practices in their organization. For example, one such statement is "We use test cases provided with candidate components as a means to evaluate them". In order to study preferences of the participants for the development practices used in their organizations we presented them with statements such as "Programming should start only after the design is completed" and asked for their agreement or disagreement with respect to their current practice and their preference. A threat to validity is that for many such questions we ask for two answers, one marked "current practice" and one marked "my preference". We initially considered splitting such items into two questions to avoid any ambiguity, for example: 'We start programming only after the design is completed' (for current practice) and 'We must start programming only after the design is completed' (for preference). However, after receiving negative feedback on the length of the questionnaire during the piloting stage, we decided to keep the total amount of text shorter, and ex-

[4] http://www.nessi-europe.com/

[5] CBSE = Component-based Software Engineering

plain the approach explicitly in the introduction to the survey. While we cannot rule out the possibility that some responses may have been distorted as a result of the shortened text, we consider the probability of it as low. In addition, some observations by others were also considered during the design of the current study. Turk and colleagues [14] has identified several limitations of agile methods when applied to building reusable artefacts. These limitations are about documentation, quality assurance process, general development and continuous redesign. They are discussed later in the section regarding challenges in agile component development. Several project characteristics that are likely to hinder the straightforward adoption of agile methods are identified in the works of Kruchten [15] and Boehm and Turner [16]. They consider project size, criticality, age of the system, rate of change, business model, team distribution and project type.

In order to study the differences between current practices and preferences as stated by the practitioners, we have applied some statistical methods. We identified 33 practices from various software process models (e.g. waterfall, agile, adaptive etc.) with some redundancy built in to address reliability. We used software process literature and experience reports to identify these practices (e.g. [17-23]). Participants in the survey were asked how strongly they agreed or disagreed with those items in terms of current practices in their organization and their preferences. We then conducted paired samples t-tests (alpha = 0.05) to evaluate any significant differences between the current practices and preferred practices for each of the items. Results regarding agile practices are presented in this paper.

4 Challenges for Agile Development of Components

By considering previous research [14-16], and [3] this section is divided into four subsections which discuss the responses in the light of each of the identified main challenges. The four challenges and how they are solved in practice or could be solved in the opinion of industry professionals who filled the questionnaire are described in the following subsections.

4.1 Challenge 1: Receiving Feedback

Components are often built to be reusable units and to satisfy the needs of many different customers, systems and contexts. Agile development seeks to receive early feedback from different sources [24] but it is a challenge to require and respond to the feedback of the various customers of the component and integration systems. 11 out of 14 of the respondents say that they deliver functionality incrementally, not only for components that are available on the market but also for components that are used internally in the company. Also, the answers show strong and very strong approval to using incremental delivery in their projects. The most commonly used delivery interval is “up to a month”, but many of the participants also responded that the interval greatly varies.

All of the respondents that are involved in development of components that are available on the market (OTS components) answered that they use different mechanisms to receive feedback from the various customers of their component. Most of them give access to customers to the early versions of the component. However, minority are those that give the end customers access to component source code without permissions to change or reuse it.

4.2 Challenge 2: Involving a Real Customer

For some types of components, such as COTS, there are many end customers, which makes it impossible to involve the real customer as suggested by agile principles. Typically some business or marketing experts need to act as proxy customers [3]. Also, we are aware of one COTS vendor which presents the current state to their key customers in web conferences every second week, and allow interaction in these virtual meetings. According to the respondents however, in general this distance to the real customers seems to be a big challenge. Nine out of the 13 respondents in this section report that inefficient communication mechanisms with clients or business people have great impact on delaying the deliveries of their project: in a eight-point scale from 0 to 7 (0 = no impact; 7 = very strong impact), they have chosen 5, 6 and 7 as answers. Just two of the respondents have disagreed with the statement that in their current practice developers only interact with customers or business people for capturing the requirements at the beginning of the project and then for acceptance testing at the end of the project – even in projects claiming to be agile. Although this seems to be the general state of practice today, 11 of the respondents say that management should encourage regular interaction between developers and customers or business people. This seems to be a problem whose solution is not adopted yet.

4.3 Challenge 3: Responding to Change

One big challenge of applying agile methodologies to component development is about responding to change [3]. When specifying and designing a component additional compatibility issues, standards compliance and interface dependencies should be considered.

In this respect, the answers received in the survey show that there is no common practice to requirements elicitation in industry. Half of the respondents have answered that they do not put extra efforts to predict and analyze future requirements for component usage in their project. The other half has given the opposite answer. In particular, both positive and negative answers are given from people in projects with relatively high rate of requirements change – the requirements are likely to change in the next couple of months. The only consistency we find is that all respondents claiming to use an agile method say that they do not try to predict the future requirements for component usage.

All respondents have answered that if enough efforts for building good and maintainable design of a component are not spent in advance, the cost of change for a component is really high. However, eight of the respondents show a preference towards starting programming before design is completed. There need not be any big conflict in this: programming may often be a good way to explore the design space, for example prototyping different architectures or algorithms. For the projects with high risk and life-critical systems the respondents have answered that programming should start only when design is completed, as is the usual standard practice in those, less agile domains [25].

In addition, another challenge identified by Turk et al is in applying continuous redesign to reusable components [14]. However, our data shows that people from industry are not afraid of redesigning a reusable component provided that component interfaces are kept. Even though in practice they redesign their components, more interesting is the fact that they have shown a strong opinion towards introducing more changes to component design and code. 10 out of 13 of the respondents say that regular changes to working code should be encouraged if they improve the code. Also, most of them disagree that a piece of code should rarely be modified once it starts working.

4.4 Challenge 4: Amount of Verification and Documentation

There are two specifics of component development that add extra complexity to the verification process of a component. The first one comes from the fact that components are usually verified without a context. The second is reusability of a component which involves more complex and formal verification [3] and [14]. For subcontracting, all respondents state that there are explicit demands on verification and documentation. Moreover, the system builders say that in general they are satisfied with the amount of component verification performed, and that verification of the system is made easier because components are verified separately. For Off-the-Shelf (OTS) component users, the opinions differ on whether verification that has been performed is enough, and whether system verification is made easier. It is not possible to discern any difference between domains, so our conclusion is that verification and/or documentation of some OTS components is insufficient, and/or that the expectations of some system builders are unrealistic.

For most of the components that are available on the market (OTS components) the component developers among the respondents state that they certify their components. In addition, all of them answered that they equip the components with suites of tests. More than half of component developers think that extensive documentation, extensive verification of system functionality and performance, and formal reviews and inspections do not have significant impact on delaying the deliveries of their project. At the same time all of them automate their testing process as much a possible and that they use to a great extend many testing techniques. These observations show that the current component verification process is efficient even though additional complexity and formality is involved. The differences in the verification process of a component seem to be dependent on common factors such as professional skills and the criticality of the project rather than anything particular to the fact that a component is verified in isolation.

5 Challenges for Agile Development of Component-Based Systems

The current section presents the challenges that component users could face when they try to adopt agile development. The section is divided into two subsections which discuss the responses regarding previously identified challenges [3]. The two challenges and how they are addressed in practice are described in the following subsections. When analyzing the data regarding component users, it is important to consider the types of components used. All projects responding to the survey combine components built in-house and outside, but the distribution among the organizations differs between components built on subcontracting agreement and general OTS.

5.1 Challenge 1: Design Lock-in vs. Responding to Change

A previously identified challenge in introducing agile principles and practices to the development of system based on components, especially OTS, is the conflict between the lock-in embedded in the choice of a pre-existing, reusable component on one hand, and being responsive to change and continuously redesign the system on the other. Almost all of the respondents agreed that redesigning a system is difficult when building a system out of components (more than otherwise). Furthermore, there is no difference in the opinions of practitioners that are using mostly in-house component to ones using OTS components. The conclusion is that the division into components, once decided, is cemented into the design and fundamentally affects further design decisions. This interpretation is further supported by the fact that the respondents who do not think system redesign is very complex are using just a small number of components (up to 10) when building their systems. It seems therefore inherently contradictory to be agile in this sense when building large systems.

5.2 Challenge 2: Comprehensive Requirements Specification

Another challenge that has been identified previously [3] and supported by the opinion of industry participants of the survey is the difficulty of changing (some of) the requirements of a system built out of components. The rationale behind this fact is that the previous selection of a component may be invalidated by the changes and the selection of a replacing component comes with high additional costs. In support, the questionnaire respondents prefer to predict and comprehensively specify the system's requirements. We have previously suggested that by introducing the agile practices of prototyping and customer involvement during the component selection and evaluation process the impact of this contradiction can be mitigated. Furthermore, the development process will benefit from applying test-driven approach, including the component selection and evaluation process.

One third of the respondents answer that they don't involve customers or business people during their component selection and evaluation process. A majority of projects following agile methodologies are doing so. Respondents who involve customers or business people in their component selection and evaluation process also use intensively prototyping to clarify requirements. Prototyping as a means for clarifying requirements is used by a great majority of respondents. Just one tenth of them answer that they never introduce prototyping for clarification of requirements. Prototyping has almost the same usage levels when applied for evaluation of component assemblies. More than three fourths of the respondents answer that they intensively use prototyping for examination of technology or architecture.

Just one of the respondents of the survey have stated negative opinion to the statement that using tests to evaluate candidate components is more efficient than reading comprehensive documents describing component behaviour. Further, the idea is supported by the fact that component developers who filled our survey respond that they equip the components with suites of tests. However, currently just one fourth of system builders evaluate their components with tests. They use both test cases provided with candidate components and the one they cerate to evaluate components. Moreover, almost all respondents use the tests created during component selection process further as integration tests.

6 Agile Practices in CBSE: Preferences of Practitioners

In the previous sections we discussed how the answers of the practitioners who both develop components and integrate components relate to identified challenges. In some cases solutions for them were provided. In this section we will describe how development process can be improved by adopting some agile practices. This is shown by investigating the differences between current practices used in industry and preferences of practitioners towards using agile practices in their projects.

Table 1. Process practices among Component Developers and Component Users

	Description	Component Developers		Component Users	
		Preferences compared to current practices	Significance (p-value)	Preferences compared to current practices	Significance (p-value)
1	Management should encourage regular interaction between developers and customers/business people	↑	.034	↑	.001
2	Management should provide flexibility for employees to form project teams	↑	.012	↑	.001
3	There should be general guidelines and principles for software development but not detailed rules	↑	.031	↑	.001
4	Comprehensive documentation should be an essential part of software development			↑	.008
5	The main focus of the team should be on the production of all artefacts (e.g. design documents) not just code			↑	.014
6	Delivering software to customers should be done incrementally			↑	.001
7	Designing and coding should be done incrementally	↑	.018	↑	.001
8	Project members should frequently meet to update each other on progress of the project	↑	.008	↑	.001
9	Project planning should be incremental, one iteration at a time	↑	.095	↑	.003
10	How far a project has progressed should be determined by the phase the project is in (e.g. requirements phase, design phase, etc.)	↓	.012	↓	.007
11	Regular changes to working code should be encouraged if they improve the code in some way (e.g. its design, its structure etc.)	↑	.032	↑	.03
12	Procedures and processes should be allowed to be changed often if the change brings in an improvement			↑	.026
13	Test cases should be written before writing code			↑	.001
14	Testing and code development should not be distinct phases in a project			↑	.008
15	Organisations should be hierarchically structured	↓	.02		
16	Breaking rules and procedures should be all right in order to get things done	↓	.047		
17	Before we start designing or coding we should try to elicit requirements as comprehensively as possible			↑	.02

Table 1 shows the practices and opinions being surveyed for which there is a statistically significant difference between current practices and preferences of IT practitioners (for significant items, the p-value is shown in the table; the lower the p-value, the more significant the result). Comparing the mean values of the responses allows us to determine whether the current practice needs to be strengthened or lightened. This is indicated by the direction of the arrows in Table 1 (an up arrow means that the respondents prefer the practice to be strengthened compared to the current state, and a down arrow means that the respondents prefer the current practice to be lightened).

Investigating the differences between current practices used by component developers and their preferences towards using agile practices in their projects, we found that in the CBSE community, there were significant differences in the case of 10 of the 33 practices we tested. All of these 10 practices are connected to agile development. In summary, component developers prefer less rigidity than what is available to them currently. This is indicated by an increased preference for interaction with customers, more frequent meetings, flexibility in identifying teams, a preference for general guidelines and principles rather than rigid rules, a preference for less hierarchical organizational structures and a preference for incremental planning and coding activities. On the other hand, their desire for flexibility does not extend to the breaking of rules to get things done, which is a positive response in an ethical sense. It is also worth noting that all of the above practices identified follow the agile principles. The one practice that is perhaps not emphasised in the agile philosophy, but identified as a preference by the component development community is the desire to see more emphasis on update of all related documents when a requirement, design or code is changed.

The community of component users employs a variety of processes and therefore their preferences are more varied than in the case of component developers. As shown in Table 1, there are differences in 15 of the 33 practices we tested, and not all of them are related to agile development However, surprisingly, there is commonality between the component users and developers in eight of the practices which were all agile. The study indicates that people in both organizations developing components and organizations using components prefer less rigid process practices than what are available to them currently. In the case of component users, factors such as the domain where integration happens probably are influencing their preferences.

7 Summary and Conclusion

We have reported a number of findings from an empirical study, by means of a web-based survey, of how agile practices and principles are applied in the development of software components and component-based systems. We conclude that components are often delivered incrementally, and component developers implement mechanisms to receive feedback from component users, even though there is not one single customer. The agile principle of having a customer involved in the project is evidently a big challenge, and the current practice is to interact mainly with internal proxy customers, and mostly at the beginning and end of the development, with the result of delayed deliveries. The agile principle of responding to constant change is in conflict with the necessity of preserving component interfaces, and the study shows that components need to be well designed in advance in order to facilitate further changes in requirements. Component boundaries often represent organizational boundaries, legal contracts, and unsynchronized processes, and so the verification and documentation of components must be more explicit and formalized than usually suggested by agile methods. This is also supported by the fact that not so many component integrators use the tests provided with candidate components as a means to select and evaluate them. However, the opinion stated by practitioners is that using tests to evaluate candidate components is more efficient than reading comprehensive documents describing component behaviour.

The study also strongly indicates that people in both organizations developing components and organizations using components prefer less rigid process practices than what are available to them currently. Thus, while practitioners, presumably on good grounds, prefer less rigidity, and the component-based paradigm potentially promises higher quality, lower costs, and shorter time-to-market, the combination of the agile and component-based paradigms require special attention.

For all empirical research with limited number of participants, we would like to further validate the findings of this study. Also as future work we want to investigate more deeply possibilities for applying test-driven approach to component selection and integration and define a development process based extensively on tests. We would also like to make some pilot deployment of the process in industrial projects.

Acknowledgments

This work was partially supported by the Swedish Foundation for Strategic Research (SSF) via the strategic research centre PROGRESS, and the Bulgarian Ministry of Education and Science. The authors also want to express their thanks to all the questionnaire respondents and the people who have been involved in earlier phases of this research. Also, the authors want to thank the anonymous reviewers for their valuable feedback which helped improve our knowledge and this paper.

8 Literature

1. Crnković, I., Chaudron, M., Larsson, S.: Component-based Development Process and Component Lifecycle. In : International Conference on Software Engineering Advances (ICSEA'06), Tahiti (2006)
2. Krasteva, I., Branger, P., Land, R.: A Systematic Comparison of Agile Principles and the Fundaments of Component-Based Software Development., Västerås (2007)
3. Krasteva, I., Branger, P., Land, R.: Challenges for Agile Development of COTS Components and COTS-Based Systems – A Theoretical Examination., Funchal, Portugal (2008)
4. Cockburn, A.: Crystal Clear: A Human-Powered Methodology for Small Teams. Addison-Wesley Professional (2004)
5. Manhart, P., Schneider, K.: Breaking the Ice for Agile Development of Embedded Software: An Industry Experience Report. In : International Conference on Software Engineering (2004)
6. Bowers, J., May, J., Melander, E., Baarman, M., Ayoob, A.: Tailoring XP for large system mission critical software development. In : XP/Agile Universe (2002)
7. Wayrynen, J., Boden, M., Bostrom, G.: Security engineering and extreme programming: an impossible marriage? In : Conference on Extreme Programming and Agile Methods (2004)
8. H., Glazer, J., Dalton, D., Anderson, M., Konrad, S., Shrum: CMMI® or Agile: Why Not Embrace Both! (2008)
9. Albert, C., Brownsword, L.: Evolutionary Process for Integrating COTS-Based Systems (EPIC). (2002)
10. Cooper, Kendra: Can Agility be Introduced into Requirements Engineering for COTS Component Based Development? In : International Workshop on Software Product Management (IWSPM) (2006)
11. Navarrete, F., Botella, P., Franch, X.: How Agile COTS Selection Methods are (and can be)? In : Euromicro Conference on Software Engineering and Advanced Applications (SEAA) (2005)
12. Scotto, M., Sillitti, A., Succi, G., Vernazza, Tullio: Agile Methods. In : CBSE: state of the art, practices, and future directions. World Scientific ((to appear))
13. Causević, A., Krasteva, I., Land, R., Sajeev, A., Sundmark, Daniel: An Industrial Survey on Software Process Practices, Preferences and Methods. (2009)
14. Turk, D., France, R., Rumpe, B.: Assumptions Underlying Agile Software-Development Processes. Journal of Database Management (2005)
15. Kruchten, P.: Situated Agility. In : International Conference on Agile Processes in Software Engineering and Extreme Programming, Limerick, Ireland (2008)
16. Boehm, B., Turner, R.: Balancing Agility and Discipline: A Guide for the Perplexed. Addison-Wesley (2004)
17. Henninger, S.: Tool Support for Iterative, Adaptable and Agile Software Development Methodologies. In : ICSE Workshop on Iterative, Adaptive and Agile Processes (2002)
18. Beck, K., Beedle, M., van Bennekum, A., Cockburn, Alistair, Cunningham, Ward, Fowler, Martin, Grenning, J., Highsmith, Jim, Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D.: Manifesto for Agile Software Development. Available at: http://agilemanifesto.org/
19. Cockburn, A., Highsmith, J.: Agile Software Development: The People Factor. IEEE Computer 34(11), 131-133 (2001)
20. Inchaiwong, L., Sajeev, A., Huang, X., Ramingwong, S.: A Comparison of Software Process Models. In : International Conference on Software Engineering Research and Practice (2008)
21. Boehm, B.: Get Ready for Agile Methods, With Care. IEEE Computer 35(1) (2002)
22. Abrahamsson, P., Salo, O., Ronkainen, J., Warsta, J.: Agile Software Development Methods: Review and Analysis. (2002)
23. Nandakumar, J., Avisson, D.: The Fiction of Methodological Development: A Field Study of Information Systems Development. Information Technology and People 12(2) (1999)
24. Beck, K.: EXtreme Programming EXplained: Embrace Change. Addison Wesley (1999)
25. International Electrotechnical Commission (IEC): Functional safety of E/E/PE safety-related systems. (1998)
26. Land, R., Blankers, L., Chaudron, M., Crnković, I.: COTS Selection Best Practices in Literature and in Industry. In : Proceedings of 10th International Conference on Software Reuse (ICSR), Beijing, China (2008)

9 Author CVs

Iva Krasteva

Iva Krasteva is a PhD Student at Sofia University „St. Kliment Ohridski“, Bulgaria. Her research interests are in the field of software processes, agile methodologies for software development, component-based development and SOA architectures. Her current studies are focused on factors and mechanisms for successful adoption of agile methodologies in different domains, environments and projects.

Iva is working as Quality Assurance Engineer at Rila Solutions EAD since 2004.

Rikard Land

Rikard is a senior lecturer at Mälardalen University. He has developed both database applications as well as safety-critical software, at ABB, Westinghouse, and CC Systems. His research interests are software processes and practices and component-based software engineering, especially for the domain of distributed embedded systems. He received his M.Sc. from Mälardalen University in 2001, and his Ph.D. with his thesis on software systems in-house integration in 2006.

A. S. M. Sajeev

A. S. M. Sajeev is a Professor of Computer Science at the University of New England, Australia. His main research interests are in Software Engineering: Software Processes, Metrics, Testing and Project Management.

He holds a PhD in Computer Science from Monash University, Australia.

Professor Sajeev is a Fellow of the Institution of Engineers, Australia.

Agile Software Development in Distributed Environments

Kerstin V. Siakas, Errikos Siakas

Alexander Technological Educational Institution of Thessaloniki,
Department of Informatics, P.O.Box. 141, GR-57400, Greece
Email: siaka@it.teithe.gr, esiakas@gmail.com

Abstract

The software industry has tried to find solutions to the persisting software crisis (an inability to develop software on time, on budget, and within requirements). New methodologies for dealing with the increasing complexity of systems and the continuously changing environment, both in technology and customer requirements has created a rapidly rising interest in new approaches such as agile methodologies. Many companies have adapted, tailored and customised the agile processes to fit their own organisational practices and culture. Simultaneously a trend towards the globalization of business in general and of software-intensive sectors in particular has emerged. Access to world-class software professionals, improved quality and time-to-market to lower costs has triggered several companies to explore these new worldwide business relationships. As a result of these two main trends in today's software development there is a huge interest in the possibilities to blend the two approaches. However, basically both rely on totally different philosophies and include many different challenges on their own. This paper aims to identify success factors and challenges for agile distributed software development by applying an extensive literature review. The challenges are unfolded and best practices for distributed agile development are proposed.

Keywords

Outsourcing, distributed software development, agile practices, agile software development

1. Introduction

In today's highly competitive and rapidly changing global environment offshore outsourcing (software development taking place by service provider outside the national border of the customer country) and distributed software development in different geographical distances have become a common business reality. Rapid advances in Information and Communications Technologies (ICTs) have provided an infrastructure that facilitates globalisation and global software development. Advantages, such as access to world-class software professionals, improved quality and productivity, costs effectiveness and shorter time-to-market promote a dynamic global business environment. However, the increased complexity of international organisations and worldwide business relationships exacerbated with intensified competition offer many challenges, mainly within human and cultural issues (Concuir et al., 2006). The globalisation of the software market has also changed the contextual boundaries of Information Systems (IS) research and practices to include the wider societal context (Siakas and Balstrup, 2006).

On the other hand the software industry has tried to find solutions to the persisting software crisis, defined as an inability to develop software on time, on budget, and within requirements (Boehm, 1981). A myriad of systems development methodologies have been proposed and used to address the problems of ambiguous user requirements, non-ambitious systems design, unmet deadlines, exceeded budgets, poor quality software with numerous 'bugs' and poor documentation. These and other factors made Information Systems inflexible to future changes and difficult to maintain (Berki et al., 2006). Recently there is a growing interest in new approaches such as agile methodologies, which many companies have adapted, tailored and customised to fit their own organisational practices and culture. Apart from the quality properties of implementability, changeability and testability, agile methodologies have re-introduced a stakeholder- or user-centred approach relating to requirements conformance, considering users as co-responsible for ensuring that the application fits the purpose. In order to ensure competitive advantage, end-user acceptance and general stakeholder satisfaction is considered key-issues. Agile approaches concentrate on the use of computational principles combined with lightweight methods supporting continuous change and the software/information products' frequently released versions.

As a result of these two main trends in today's software development there is a huge interest in the possibilities of blending the two approaches, which basically rely on totally different philosophies. Distributed software development relies on formal processes opposed to agile development, which relies on informal processes. The challenges posed by business stakeholders seem to be how to blend the two approaches for increased Return on Investment (ROI) and business value.

The motivation behind this research is to identify success factors and challenges of both agile and global software development on their own and applied together in order to understand how agile distributed software development can successfully operate across national and cultural boundaries.

In this paper we address, by undertaking a comprehensive literature review, the integration of contemporary software development concepts applied to global business processing and outsourcing efforts. The dynamics that bear on the success of agile distributed software development are made explicit and the implications and challenges of a blended approach are unfolded. As a result a set of best practices and guidelines for improved awareness of risks and ways of avoiding pitfalls are proposed.

2. Outsourcing

The evolution of the internet has endorsed organisations to establish business partnerships beyond geographical boundaries. Global software development, including outsourcing, subcontracting and partnerships, is becoming increasingly common (Paasivaara and Lassenius, 2006). Outsourcing is a contractual relationship where an external organisation (a third party service provider) takes responsibility for performing all or part of an agency's (contractor's / client's) IT/IS functions at a pre-determined price and according to predetermined performance criteria (Siakas and Siakas, 2006a). The contract aspect and the responsibility aspect are central issues. Companies may have multiple sourcing relationships in different countries and a service provider can simultaneously be a client to another service provider.

The primary motivation for outsourcing is cost-effectiveness. By outsourcing non-core activities organisations can focus on broader business concerns and redirect resources towards research, development and activities that provide a greater return. Simultaneously organisations, by having operations accomplished by outside service providers (who usually are experts in the field) gain access to world class capabilities, such as new technologies, tools, methodologies and procedures that the organisation may not currently possess. Improved business focus, flexible adoption of best practices and increased availability of diverse skills to reduced labour costs are drivers that will enable customer companies to achieve faster, more efficient and more economical business processes and subsequently to increase their competitive advantage.

When companies outsource they become more flexible, more dynamic and more able to meet the changing opportunities (Siakas and Balstrup, 2006). Outsourcing also provides opportunities for sharing risks across many companies. Service providers make investments not only for their own company but on behalf of their many contractors/customers. By sharing these investments the risks are significantly reduced. Outsourcing can also be a good solution for organisations that lack required resources for applications needed to be developed or modified. However, organisations should be aware of the fact that outsourcing can cause dysfunctional organisations due to loss of control and in-house expertise, cultural differences and dilution of the company knowledge base when outsourcing (Siakas and Siakas, 2006b).

The countries involved as customers / clients are mainly North America and Europe with Japan following (Imsland, 2003). The prevailing software service provider country is India, dominating 80–90 percentage of the total offshore development revenue worldwide (Khan et al., 2003). According to a survey by Ware (2003) including 101 IT professionals, the benefits of offshore outsourcing were considered as following: lower cost (78%), increased IT department productivity (44%), reduced project timeline (37%), competitive advantage (30%) and internal customer satisfaction (20%). On the other hand the challenges of offshore outsourcing were: managing communication (67%), cultural differences (51%), lack of internal processes for specifying work (40%) and lack of internal customer management skills (32%). It seems that communication and cultural differences are significant uncertainty factors difficult to control and manage. Cooperation between outsourcing partners and members of project teams looms as an important factor for success (Cleland and Gareis, 1994). Communication and trust are at the heart of cooperation and the slightest cultural misunderstanding can create serious damage in the outsourcing relationship (Siakas and Siakas, 2008; Siakas et al., 2006). Research regarding virtual / distributed teams has also verified that the main differences between collated teams (team-members working in the same office) and distributed teams (team-members working on the same project in different locations) are within communications and trust (Balstrup, 2004). Language, time and distance are factors that can lead to inadequate communication and subsequently to difficulties in building trust. Also ignorance of cultural differences creates suspicions and distrust. Despite the unprecedented development of Information and Communication Technologies (ICTs), which have facilitated distributed communication and collaboration and subsequently easier access to outsourcing partners, there are factors which must not be neglected. Potential for increased ambiguity, complexity and confusion usually occurs in situations where a single agreement has to be taken or when overall procedures have to be developed. Managers seem to be worried about risks and their own capabilities in order to be successful in an increasingly complex global context. The effective management of cultural diversity in a global context is a challenge and a competitive advantage. Managers who are involved in cross-cultural communications and negotiations need to develop characteristics such as cultural sensitivity, flexibility and adaptability.

To help service contractors / clients to search, select and collaborate with service providers the eSourcing Capability Model for Client Organizations (eSCM-CL, released in February 2006), a best practices capability model from the clients' perspective, was developed (Hefley and Loesche, 2006). In order to support service providers to improve their capabilities the eSourcing Capability Model for Service Providers (eSCM-SP, released in April 2004) was launched (Hyder et al. 2002, 2004a, 2004b). Both models are aligned and contain 84 best practices. Each practice is organised into three dimensions, namely sourcing life-cycle, capability area and capability level. The eSCM-CL is expanded with activities the clients need to perform, such as deciding an overall outsourcing strategy and prior internal alignment, before the beginning of the outsourcing relationship. The purposes of the models are to give service providers guidance to help them improve their capabilities across the sourcing life-cycle and to support risk reduction activities, to provide clients with objective means of evaluating the capabilities of service providers and to offer clients and service providers a standard that they can use when they want to differentiate them-selves from their competitors.

Another important tool for assessing the cultural fit between a service provider and a contractor in an outsourcing business partnership is the SQM-CODE (Software Quality Management – Cultural and Organisational Diversity Evaluation) (Siakas and Hyvärinen, 2006). The SQM-CODE tool can be used in any global context, e.g. in an organisation wanting to expand to a new geographical area and/or in virtual organisations for assessing teams working across organisational boundaries. The aims of the SQM-CODE model are to assess the fit between national and organisational culture and subsequently to help these organisations in developing cultural sensitivity and to predict a suitable Software Quality Management system in the area of interest.

Another contemporary approach that has been of interest to IS researchers for quite a while is how to manage knowledge in software development (Kjaergaard et al., 2008). This also counts for outsourcing organisations that need to manage knowledge without diluting the company secrets (Siakas and Balstrup, 2006). On the whole organizations try to gain business advantage by using Knowledge Creation processes (KC) in order to "capture" knowledge and use it to make wiser decisions about strategy, competition, products, production and service life cycles (Davenport and Prusak 1998), as well as to improve its effort in today's very competitive and uncertain environment. Organisational Knowledge is created by an organizationally specified systematic process for acquiring, organising and communicating both tacit and explicit knowledge of employees so that other employees may make use of it in order to be more effective and productive (Alavi and Leinder, 2001). This experience is documented and stored in a Knowledge Management System (KMS) preparing the organisation to react on the future, based on the knowledge that is acquired from its own organisational experience. Knowledge development is a part of organisational learning. Learning is considered to be more important in creating sustainable competitive advantage because it deals with the process of developing organisational capability (Tippins and Sohi, 2003). Organizational learning consists of four components: information acquisition, information dissemination, shared interpretation, and development of Organisational Memory (OM). KM and OM are concepts well known from organisational science and learning theory (Mayer and Remus, 2002). How about outsourcing relationships? How much know-how and knowledge is the client ready to transfer to the service provider without risking the service provider to become a future competitor?

Different frameworks have been developed to characterise the knowledge management cycle in organisations (Demarest, 1997; Ruggles, 1998). The major activities in the cycle include identification, generation, codification and transfer. The preliminary stages of identification and generation are concerned with the acquisition of knowledge. Sources of knowledge could be internal or external, including experienced employees, experts and different reports. The relevant knowledge is identified and generated through techniques such as discussion groups, presentations and shared workspaces. The generation of process knowledge differs from the generation of tacit knowledge held by employees. The latter is generated through discussion groups, online conferencing etc., while explicit process knowledge is generated as a result of conscious management and monitoring of processes (Amaravadi and Lee, 2005). Organisations depend on process improvements for productivity increases and are continuously fine tuning the parameters of the process, such as quality and efficiency.

3. Agile Development

The new trend in software development is the agile approach. According to a global survey (722 respondents with average of 2.3 years of experiences of agile development) the respondents considered that the value they actually realised from implementing agile development had improved or significantly improved the following issues (Agile Development Survey, 2006): enhanced ability to manage changing priorities (92%); increased productivity (75%); improved team morale (74%); enhanced software quality (74%); accelerated time-to-market (72%); reduced project risk (72%); alignment between IT and business goals (66%). As we can see from the top response and also a key-factor in agile development is the anticipation of changing requirements development. The basis of agile development lies in small teams working in co-located development environment developing non-safety critical software (Abrahamsson, 2005). Co-location reduces risk (Norton, 2008). In a recent survey comparing agile projects it was found that success rates for co-located teams were over 20% higher than for geographically distributed teams (Ambler, 2008). Also Lindvall et al. (2004) argue that while agile practices can match the needs of large organizations, in particularly for small co-located teams, integrating new practices with existing processes and quality systems require further tailoring to integrate agile practices with existing established structures and processes.

The fundamentals of agile practices comprise flexibility and quick response to changing requirements, which are considered as a necessity to sustain and improve customers' competitive advantage. Agile practices promise effective and successful software development without the cost of a heavy quality system, which requires numerous intermediate work products and rigid standard procedures and heavy documentation (Siakas et al., 2005). Agile methodologies emphasise user satisfaction through user participation, recognition of and response to continuous changing requirements and frequent delivery of products together with adaptive and iterative software development by self-organising teams that recognise that team-members are committed competent professionals who are able to choose and follow an adaptive process (Siakas and Siakas, 2007).

Current practices of agile development presume the customer to be co-located and prepared to invest in coaching the software development team in the business context or application domain. Elicitation and management of volatile requirements are facilitated by promoting collaboration and communication (Berki et. al, 2006; Prior and Keenan, 2005). The on-site customer / customer representative, also called product owner (Miller, 2008) helps the team to understand customers' requirements. Customer gives frequent feedback on the deliverables from the team's last iteration. The literature has suggested that higher customer involvement results in higher quality, especially in terms of meeting requirements, implementability, changeability and testability (Siakas and Siakas, 2007). Factors, though, that affect the quality of software (and their interconnected nature) need to be identified and controlled (Georgiadou et al., 2003) to ensure predictable and measurable totality of features and quality characteristics of the particular software.

Figure 1 show the conflicts between organisational culture embracing bureaucratic plan-driven process oriented software quality management and the lightweight agile culture emphasizing customer short development cycles.

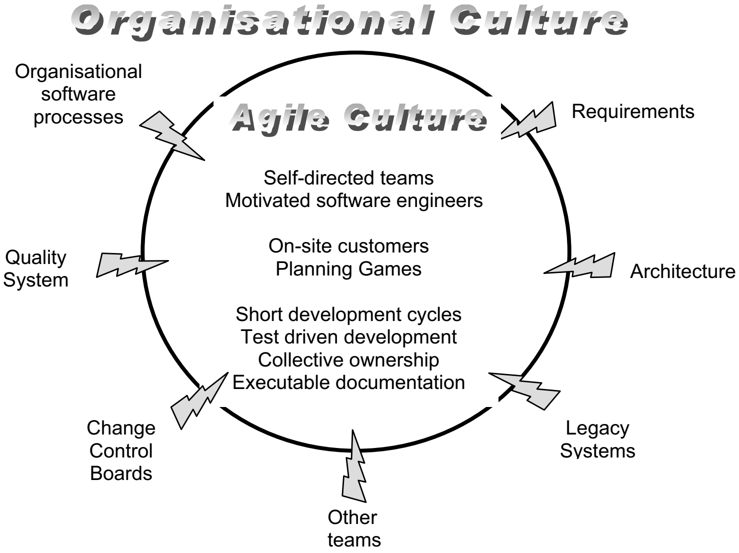


Figure 1: Aligning the organisational culture with the agile culture

Glazer et al (2008) are convinced that agile methods and CMMI successfully integrated bring substantial benefits to both agile and traditional software development organizations. Large organisations wanting to introduce agile development need to resolve incompatibilities and align the organisational culture with the agile culture (Siakas and Siakas, 2007). Figure 1 show the factors that need to be taken into consideration and aligned for avoiding conflicts. Large organisations may for example follow defined software processes, which ma result in double work when new practices are introduced (Lindvall et al., 2004). Also in traditional plan-driven process oriented software development quality is designed from the beginning into the process and subsequently into the product. In agile development the continuous refactoring and the frequent changes is sometimes seen as a risk factor (Lindvall et al., 2004).

Research has also shown that there is a visible conflict related to the amount of documentation that should be kept (Karlström and Runeson, 2005). Agile development keeps documentation to a minimum, whilst the emphasis in traditional plan driven software development is on process description and documentation of every step according to Deming's p-d-c-a (plan-do-check-act) circle (Deming, 1986). For decades the software industry has relied on these premises, through the use of ISO-9001:2000 (ISO, 2006), and Capability Maturity Models, such as CMMI (2006), Bootstrap (Kuvaja et al., 1994) and SPICE (ISO-15504) (Dorling, 1993). A balance between how much work to put into documentation and the usefulness of documentation has to be found. Many service providers (particularly in India) are CMMI certified and in the future this may be a requirement for obtaining outsourcing contracts (Biro et al. 2003). Agile development on the contrary relies on flexible processes. A compromise between the two approaches describing basic processes but leaving space for adaptation according to the situation always with emphasis on embracing customer needs is required.

4. Distributed Agile Development

Distributed agile development comprises two trends in software development, namely distributed software development (virtual teams and outsourcing) and agile development (Ramesh et al., 2006). Global agile development is a combination of traditional plan-driven process oriented software development methods, established in mainly large organisations practicing outsourcing and development processes, involving small co-located teams. The distributed team can include distributed development teams (more than one development team, each in a different location) and customers not co-located with developers and dispersed teams (no one on the team is in the same location). The utilization of globally distributed agile teams has the potential to significantly impact the field of software development (Sharp and Ryan, 2008).

4.1 Reasons for distributed agile development

Both small and large organizations have shown interest in agile methods. The main reason is that they seek alternatives to the traditional software development methodologies, which they consider too bureaucratic, and inflexible. They also feel pressure to increase productivity at lower cost while maintaining or improving quality (Lindvall et al. 2004). Business trend today also seem to push organisations into global markets, despite the uncertainties in global business environments. The business advantages of proximity to the market, including knowledge of customers and local conditions, as well as local investment seem to be important reasons for distributed software development (Herbsleb and Moitra, 2001). Lee et al. (2006) identified three components of agility (ability to adapt to changing environments) in global distributed system development, namely:

- Agile IT strategy (flexible IT strategies to meet specific local business needs);
- Agile IT infrastructure (an IT platform amendable to support local business needs, and collaboration, communication, coordination and control);
- Agile IT project management (flexible management skills to adapt to local and cultural convergence values)

If we consider agility of software development we can agree that also agile teams need to go global in order to be competitive, due to the following reasons (Miller, 2008; Moore and Barnett, 2004).

- *Global markets:* As businesses expand into new markets expertise is needed in those markets through mergers, acquisitions and subsidiaries located in these markets;
- *Reducing costs:* Companies often seek to reduce costs through outsourcing to regions with cheaper salaries;
- *Global talent:* Specialized advanced technical expertise may not be available in the location of the company, but may be found in other markets.

Some traditional management tools, such as Capability Maturity Model Integrated (CMMI) or eSourcing Capability Maturity Model for Service Providers (eSCM-SP) and/or Clients (eSCM-CL) may be used to provide solid software development (Siakas and Siakas, 2006a), but on the same time management embracing agility is needed to address changing requirements and foster communication and interaction among clients and onshore as well as offshore development teams. Both approaches suggest different challenges on their own and in combination they offer a competitive advantage to the companies able to reap the benefits of both and to capitalize on the challenges. Also the configuration of the global agile team is suggested to be important (Sharp and Ryan, 2008). The dimensions of the configuration are proposed to be team structure, team agility and virtualness.

4.2 Success factors in distributed agile development

Three important success factors have been identified in distributed agile development, namely culture, people and communication (CEInformant, 2006). However, these factors can also be failure factors if not handled with caution. In addition, common goals, common systems, common processes and compatible technologies are important factors for success in distributed environments (Siakas and Balstrup, 2006). Below the factors are analysed in more detail.

Culture: The cultural dimension divides the agile teams into culturally homogeneous and heterogeneous teams. Culture is the most difficult to assess as it embraces facets like language, tradition, values, beliefs, norms and practices. The virtual leader must posses a profound understanding of the cultural differences within the team. Additionally the leader needs an employee at the distant location (cultural bridging staff) who caters for informal sharing of experiences and to whom the team have access to in the absence of the virtual leader (Krishna et al., 2004). The benefit of a local leader, who loyally exerts the chosen strategy and direction through self-management supported by a trusting delegation, is the natural transformation of the leadership into the local cultural context. In agile distributed practices the successful leaders need to acknowledge and reflect on their strengths and weaknesses. They must be motivated to continuously develop themselves and be aware that a present strength can turn into future weakness. Additionally they must be able to compensate for their own weaknesses by selecting employees or external service providers with complementary strength and empower them to take on the tasks they are more qualified to perform (Balstrup, 2004).The culture of the organization and its employees must be conductive and encouraging to the agile approach, which can be considered to be a culture of its own (Siakas and Siakas, 2007), since it has the characteristics of a group of people that differentiate themselves from others through a whole set of shared practices including visions, values, principles, ideals, practices etc. that emerge in the interaction between members of a group. The eXtreme Programming (XP) for example draw attention to four XP values, namely, communication, simplicity, feedback and courage, the underlying basis for the twelve principles which are translated into practices. These practices are the artefacts for the XP culture. Having a consistent culture is important to create consensus and agreement and to avoid culture clashes and friction in a group/team and in the whole organisation. The social characteristics of the team members are important. Employment of technically highly competent and competitive software professionals generates the basis for the creation of a strong culture.

People – trust – knowledge sharing: Teamwork is in essence a result of human interaction, but, in an environment where organisations formulate strategies for becoming global, working in a common place becomes unusual. In virtual environment team cohesion and group identity is difficult to achieve, because members seldom meet face-to-face and thus informal communication is reduced. A successful leader of a virtual team must excel in applying the right choice of communication means along with a profound knowledge of the effect of applying it. One of the strengths of team work is the exploitation of knowledge sharing and the dynamics of the team. If communication and trust are limited only partial knowledge sharing will take place and the potential competitive advantage will not be achieved. Personal connections are the most important issues to building trust. New knowledge always begins with the individual, whose self-interest determines in which informal knowledge creation processes to participate (Chen and Edgington, 2005). Knowledge sharing between and within teams need to be supported in order to enhance developers’ shared understanding of applications and business domains (Remesh et. al 2006), because individuals tend to hoard knowledge for various reasons (Bock et al., 2005); one reason being the cultural value system of the team, organisation or country. Within a single culture certain values, attitudes and behaviours are either favoured or suppressed (Siakas et al., 2003). In a cross-cultural team the dynamics are more difficult to understand and interpret. Making personal knowledge available to others is a central activity in knowledge creation. Explicit knowledge is formal and systematic, whilst tacit knowledge is highly personal (Nonaka, 1998). The constructive learning process in work-places is revealed through bottom-up knowledge creation spread from individual to individual in the socialisation process (Nonaka and Takeuchi, 1995). The use of social computing (Web 2.0 and/or Web 3.0) is considered to bring proximity in distributed environments similar to informal communication within and among co-located teams. Without trust-building mechanisms and knowledge sharing strategies distributed teams tend to create opposite competing poles and the team coherence is jeopardized. Two important factors for supporting collaboration are loyalty and commitment. The individuals of the virtual team and the leader must build a cohesive team committed to the common goal and through interdependent interaction generate group identity and create the feeling of belonging to the "we" group (Balstrup, 2004). Creation of cohesion is fragile and requires effective interpersonal leadership.

Communication and coordination: While there may be business reasons for distributing an agile team, distribution will also contribute to dysfunction of the team by reducing communication and increasing coordination. Developers not located together have very little informal, spontaneous conversation across sites. Agile teams rely on intensive person to person communication, both with the customer and within the team. The distributed team lacks the benefits of having people at hand both for formal and informal meeting, for coordination, problem solving and learning (Ågerfalk, 2005). Firms have become more adept at parsing out tasks and functions that require low levels of coordination. Continuous communication via frequent mutual visits (face-to-face contact) is of utmost importance since it generates trust and a useful bond between team members. The use of bridging staff (called ambassadors by Fowler, 2006) acting as local project leaders and being the main link between the two organisations (client and service provider) helps in building trust and smoothing cultural misunderstandings (Siakas and Balstrup, 2006). Another important requirement for building a coherent team is frequent communication and document transfer via all contemporary means, such as phone, email, chat, wiki and videoconferencing. Social computing becomes more and more important in distributed environments. Today ICT and media convergence allow for daily communication and collaboration through voice media, wiki, instant messaging, chats and tele-conferences (AbuShanab and Siakas, 2009). Also recognition of the importance of the communication language and the fact that cross-cultural training is needed both in advance and continuously should be taken into consideration (Foster, 2000). Miller (2008) Development Manager with the Microsoft's patterns & practices group has been involved in agile, distributed development approaches for the past five years. He argues that creating an effective distributed agile team is largely about compensating for the barriers to communication added by distribution. He provides five advices for communication.

- Video conferencing facilities have to be set up for easy availability;
- More formal scheduled nonverbal communication is needed for a distributed team;
- Communication needs to become an explicit part of duties on the team;
- Deliberatively involvement of remote team members;
- Meeting formats may also need to change due to the lack of opportunity to communicate outside the meetings. For example daily stand-up meetings may include some time for the team to discuss other topics.

Firms that understand how to take advantage of potential benefits of geographic dispersion, such as cultural diversity (understanding customer needs, innovation potentials and local expertise) (Siakas and Siakas, 2008, 2006) and different time zones (production increase by 'round-the-clock' development) (Herbsleb and Moitra, 2001) are likely to gain competitive advantage. This however, is a fragile issue that require good leadership. The slightest mistake can change potential success to serious failure.

Common goals – systems - processes - technology: Another important issue for agile distributed teams is the use of common systems, processes and compatible technologies (Heeks et al., 2001). Instead of strictly following pre-defined processes, praised by Software Process Improvement (SPI), or agile development practices, the distributed agile development team need to adjust the process to fit the evolving needs of the project (Ramesh et. al, 2006). Also common project management tools that focus on planning and tracking the progress of features (or user stories) with actual business value to customers need to be implemented.

In an experiment with 150 talented recently graduated software developers in Bangalor (Fowler, 2006) mixed with experienced UK and US developers (mentors) working as service providers it was found that continuous integration of work across multi site teams was important to keep team members updated about the status of the development. Effective communication channels and a web-page served as a notice board for changes carried out in the project. Visits, in the beginning and during the project were considered imperative for creating and sustaining relationships. The hardest part of introducing agile methods into an organisation was considered to be the cultural change it causes. Agile methodologies require democratic type of organisations emphasising consultation, participation, empowerment, consensus and compromises (Siakas and Siakas, 2007). Fowler noticed that *"Asian cultures reinforce deference to superiors …and getting people used to a distributed control style of management takes longer than you think".* He also recognises that documentation becomes more important with offshore development.

A project with a very complex problem domain affects even architectural choices, in the form of generating a system that is partitioned into subsystems. Architecture is important, it emerges from the principles and values of the team culture – and culture by its very nature cannot be planned; it can merely be guided. However, it is argued that agility may lead to more complex and not well-documented systems through a fragmented software development process (Boehm and Turner 2003). In addition, people must be well-trained and work in close communication with each other. With these considerations, implementing agile processes within multiple development teams across different geographies and time zones presents a unique set of challenges. Ågerfalk et al. call these differences for temporal distance and geographical distance (directional measure of the effort required for one actor to visit another at the latter's home site). The cultural differences they call socio-cultural distance (a directional measure of an actor's understanding of another actor's values and normative practices).

The contract aspect and the responsibility aspect are central issues in outsourcing. The high iteration frequency in agile development has consequences for contracts variables, such as scope, price and time and thus the contracts need to be flexible (Ceschi et al., 2005). This, in turn, may be a drawback for the customer's cost-analysis plans.

The advantages of distributed software development promise, according to Moore and Barnett (2004) faster, better and cheaper software. However, in order to be successful, teams need to resolve incompatibilities and align to the requirements both of agility and structure.

The generic guidelines that practitioners may draw from the study are presented in table 1, which shows an overview of the challenges in distributed agile software development discussed. It proposes actions and best practices identified in the literature to align and balance the challenges of distributed and agile software development.

Table 1: Guidelines for practices meeting challenges in distributed agile software development

Challenges	**Proposed Action**	**Best Practices**
Lack of team cohesion	Trust building mechanisms	• Frequent on-site visits by distributed stakeholders • Support of a cohesive team culture • Self-directed teams – decentralised decision structures
Degree of formalisation	Increased formalisation	• Common adjustable processes • Documentation of requirements at different levels of formality for understood functionality
Value diversity Hierarchy Structure	Cultural awareness	• Cross-cultural training • Use of cultural bridging staff • Alignment of national and organisational culture • Conflict handling mechanisms
Communication needs Communication Styles	Improved Communication	• Use of compatible ICT and Media Convergence tools • Increased formal scheduled nonverbal communication • Use of social computing tools (Informal communication) • Arranging of synchronised working hours • Communication training (cross-cultural training)
Knowledge sharing	Facilitation of knowledge sharing	• Use of product/process repository • Promotion of a knowledge sharing culture
Incompatibility of goals	Improved Communication	• Commonly defined milestones • Clear entry and exit criteria

Table 1 is anticipated to add to the insights and understanding of an effective organizational behaviour to support distributed agile development.

5. Conclusion

Globalisation is today a reality having created numerous of challenges for managers worldwide. Increased and improved capabilities of ICT facilitate continuous expansion of globalisation. Outsourcing and virtual collaborations prompt for cultural sensitivity, flexibility and adaptability, together with high awareness of risks and dangers due to cultural differences. Globalisation is a competitive advantage if handled in a right manner. The new trend in software development is agile development, which embrace continuously changing requirements with customers on board, face-to-face communication, trust and collaboration. Both offshore outsourcing and agile development comprise challenges on their own. Combining the two approaches require special caution and companies need to be aware of potential clashes due to cultural divergent values.

In this paper we tried to make more explicit the social and cultural dynamics that bear on the success of agile distributed software development by discussing challenges, implications and success factors. The main contribution of the paper is the overview, based on an extensive literature review and reflections on own research and experience, of the challenges, proposed actions to mitigate risk and best practices to align and balance the challenges of distributed agile software development. A key factor for success is to understand the impact of distribution on agile teams in order to mitigate the impact. Proximity and trust can only be created by frequent on-site visits (both ways and at all levels) and by using effective virtual communication channels and social computing for informal communication. Human, social and cultural aspects can either become constraints or competitive advantage in globally distributed collaboration. When firms that invest in global distributed software development are aware of cultural strengths and biases they can can take advantage of both differences and similarities through mutual cross-cultural synergy for growth and development.

Agile development and global software development are both new trends requiring changes in the traditional way of developing software and communicating with end-users / customers. Both approaches promise reduced costs. However, in terms of formality, they comprise two opposites; agile practices being flexible to volatile end-user requirements through intensive informal developer / customer interaction, light documentation and frequent releases opposed to offshore outsourcing relying on plans and formal interaction supported by a heavy quality assurance system. In both approaches management experiences difficulties when applying traditional management style; in agile development due to the power shift from the prerogative managerial elite to empowered software engineers and in offshore outsourcing due to the increased complexity of global organisations and their dependency on people with different underlying norms, values and beliefs. Effective communication systems, personal visits from both sides and the use of bridging staff will inevitable help to improve trust and collaboration. Businesses that endeavour distributed agile development by taking special caution to local human and cultural traits certainly deserve the potential added value promised by both approaches.

Further research will concentrate on combining a more systematic approach with practical evidence for the identification of the human, social and cultural aspects that are cruisial for success of distributed agile software development.

References

Abrahamsson P. (2005). Project Manager's Greetings - Agile Greetings, *Agile Newsletter Issue*: 1, 2004, p. 1

Abu-Shanab E. A., Siakas K. V., (2009). Collaborative Development of a New Postgraduate Course in ICT and Media Convergence: Experiences and Reflections, 14th *INternational Conference on Software Process Improvement - Research into Education and Training, (INSPIRE 2009),* 6th - 8th of April, 2009, Southampton Solent University, Hampshire, UK (Proceedings to be printed)

Ågerfalk P. J, Fitzgerald B., Holmström H., Lings B., Lundell B., Ó Conchúir E. (2005). A Framework for Considering Opportunities and Threats in Distributed Software Development, *the International Workshop on Distributed Software Development*, pp. 47 – 61

Agile Development Survey (2006). retrieved on 27.11.2006, http://www.infoq.com/news/versionone-Agile-survey

Alavi M., Leidner D. E. (2001). Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues, *MIS Quarterly*, 25:1, pp. 107-136

Amaravadi C. S., Lee I. (2005). The Dimensions of Process Knowledge, *Knowledge and Process Management,* Vol. 12, No. 1, pp. 65–76

Ambler S. (2008), Has Agile Peaked? *Dr. Dobbs Journal*, June 2008

Balstrup B. (2004). *Leading by Detached Involvement – Success factors enabling leadership of virtual teams,* MBA Dissertation, Henley Management College, UK

Berki E., Siakas K. V., Georgiadou E. (2006). Agile Quality or Depth of Reasoning? Applicability versus Suitability Respecting Stakeholders' Needs, (eds) Stamelos Ioannis and Sfetsos Panagiotis, *Agile Software Development Quality Assurance*, Idea Publishing

Biró M., Deák G., Ivanyos J., Messnarz R., Zámori Á. (2003). Using the eSourcing Capability Model to improve IT enabled business process outsourcing services. *EuroSPI 2003 (European Software Process Improvement Conference)*, Graz, Austria, 10-12.12.2003, pp. part III:1-16

Bock G-W., Zmud R.W., Kim Y-G., Lee J-N. (2005). Behavioural Intention Formation in Knowledge Sharing: Examining the Roles of Extrinsic Motivators, Socio Psychological Forces and Organisational Climate, *MIS Quarterly*, March, Vol. 29, No. 1, pp. 87-111

Boehm B. W. (1981). *Software Engineering Economics,* Prentice Hall

Boehm B., Turner, R., (2003). Using Risk to Balance Agile and Plan-Driven Methods, *Computer,* June, pp. 57-64

CEInformant (2006). Global Agile Development, *CEI's Approach to Successful IT Outsourcing*, Number: 04062006

Ceschi M., Sillitti A. Succi G., de Panfilis G. (2005). Project management in plan-based and Agile companies. *IEEE Software,* May/June.Vol 22, Issue 3, pp. 21-27

Chen A. N. K., Edgington T. M. (2005). Assessing Value in Organisational Knowledge Creation: Considerations for Knowledge Workers, *MIS Quarterly*, June, Vol. 29, No. 2, pp. 279-309

Cleland D. I., Gareis R.(eds) (1994). *Global Project Management Handbook*, McGraw-Hill, Inc. US

CMMI (2006). http://www.sei.cmu.edu/cmmi/, retrieved on 10.12.2006

Conchuir E. O., Holmström H., Ågerfalk P. J., Fitzgerald B. (2006). Exploring the Assumed Benefits of Global Software Development, *Proceedings of the conference on ICGSE*, pp. 159-168, IEEE press, New York

Davenport T. H. & Prusak L. (1998). *Working Knowledge: How Organisations Manage What They Know*, Boston, Harvard Business School Press

Demarest M. (1997). *Knowledge management: an introduction*. Long Range Planning, 30(3). 374–384

Deming W. E. (1986). *Out of the Crisis: Quality, Productivity and Competitive Position*, Massachusetts, USA

Dorling A. (1993). Spice: Software Process Improvement and Capability dEtermination, *Software Quality Journal,* 2, 93, pp. 209-224

Foster N. (2000). Expatriates and the impact of Cross-Cultural Training, *Human Resource Management Journal,* Vol. 10, No 3, pp. 63-78

Fowler M. (2006). Using an Agile Software Process with Offshore Development, white paper, 18 Jul 2006, available at http://www.martinfowler.com.articles.Agileoffshore.html

Georgiadou E., Siakas K., Berki E. (2003). Quality Improvement through the Identification of Controllable and Uncontrollable Factors in Software Development*, EuroSPI 2003 (European Software Process Improvement Conference),* Graz, Austria, 10-12.12.2003, pp. IX 31-45

Glazer H., Dalton J., Anderson D., Konrad M., Shrum S. (2008). CMMI® or Agile: Why Not Embrace Both! *Technical Note CMU/SEI-2008-TN-003*, November 2008

Heeks R. B., Krishna S., Nicholson B., Sahay S. (2001). Synching or Sinking, Global Software Outsourcing Relationships, *IEEE Software*, 18, 2, March/April, pp. 54-61

Hefley W. E., Loesche E. (2006). *The eSourcing Capability Maturity Model for Client Organisations (eSCM-CL),* Model Overview, v.1.0, ITSqc Working Paper Series Document, #CMU-ITSQC-WP-06-001, Draft for Public Review, Carnegie Mellon University, PittsburgImsland,

Herbsleb J. D. and Moitra D. (2001). Global Software Development, *IEEE Software,* March / April, Guest-Editors, Introduction, pp. 16-20

Hyder E. B., Heston K. M., Paulk M. C. (2004a). *The eSourcing Capability Model for Service Providers.eSCM-SP v2*, Part 1: Model Overview, CMU-ISRI-04-113, Pittsburg, PAL Carnegie Mellon University

Hyder E. B., Heston K. M., Paulk M. C. (2004b). *The eSourcing Capability Model for Service Providers. eSCM-SP* v2, Part 2: Practice Details, CMU-ISRI-04-114, PAL Carnegie Mellon University, Pittsburg,

Hyder E. B, Kumar B., Mahendra V., Siegel., Heston K. M, Gupta Rajesh, Mahaboob H., Subramanian P.. (2002). *The e-Sourcing Capability Maturity Model. eSCM-SP for IT enabled Service Providers,* v1. 1, CMU-CS-02-155, School of Computer Science, Carnegie Mellon University, Pittsburg

Imsland V. (2003). *The Role of Trust in Global Outsourcing Relationships*, Candidate Science Thesis, Oslo University, Department of Informatics

ISO (2006). http://www.iso.org/, retrieved on 5.10.2006

Karlström D., & Runeson P., (2005). Combining Agile Methods with Stage-Gate Project Management, *IEEE Software*, May/June, pp. 43-49

Kjaergaard A., Kautz K., Nielsen P. A. (2008) Making Sense of Project Management, in P-A, Nielsen, Kautz, K., *Software Processes & Knowledge, Beyond Conventional Software Process Improvement,* Software Innovation Publisher, Aalborg University, Denmark, May 2008, pp. 71-88

Khan N., Currie W. L., Weerakkody V., Desai B.((2003). Evaluating Offshore IT Outsourcing in India: Supplier and Customer Scenarios. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, January, pp. 239–248

Krishna S, Sahay S., Walsham G. (2004). Managing Cross-cultural Issues in Global Software Outsourcing*, Communications of the ACM*, April, Vol. 47, No 4

Kuvaja P., Similä J., Kranik L., Bicego A., Saukkonen S., Koch G. (1994). *Software Process Assessment and Improvement – The BOOTSTRAP Approach*, Blackwell Publishers, Cambridge, MA

Lee O-K., Banerjee P., Lim K. H., Kumar K., Hillegersberg J. V., Wei K. K. (2006) Aligning IT Components to Achieve Agility in Globally Distributer System Development, *Communications of the ACM*, Oct. 2006, Vol. 49. N0 10, pp. 49 - 54

Lindvall M., Muthig D., Danino A., Wallin C., Stupperich M., Kiefer D, May J., Kählönen T. (2004). Agile Software Development in Large Organisatios, *IEEE Computer Society*, December, pp. 26-34

Maier R., Remus U (2002). Defining Process-oriented Knowledge Management Strategies, *Knowl-edge and Process Management*, Vol. 9 No. 2, pp. 103–118

Miller A. (2008), Microsoft patterns & practices October 2008, http://www.ademiller.com/tech/, retrieved 15.02.2008

Moore S., Barnett L. (2004), “Offshore Outsourcing and Agile Development”, *Best Practices, Forrester Research,* Inc, September 20, 2004

Nonaka, I. (1998). *The Knowledge-Creating Company, Harvard Business Review on Knowledge Management,* Harvard Business School Press

Nonaka, I., Takeuchi H. (1995). *The Knowledge-Creating Company – how Japanese Companies Create the Dynamics of Innovation,* Oxford University Press, UK

Norton D. (2008), Making the Most of 'Agile' in a Distributed Development Environment, *Gartner Research G00159246*, August

Paasivaara M., Lassenius C. (2006). Could Global Software Development Benefit from Agile Methods? IEEE International Conference on Global Software Engineering (ICGSE'06)

Prior P., Keenan F. (2005). Requirements Management in a Distributed Agile Environment, Transactions on Engineering, *Computing and Technology,* v4, World Enformatika Society February, pp. 204 -207

Ramesh B., Cao L., Mohan K., Xu P. (2006). Can Distributed Software Development be Agile?, *Communications of the ACM*, Oct. Vol. 49, No.10, pp. 41 - 46

Ruggles R. (1998). The state of the notion: knowledge management in practice, *California Manage-ment Review*, Special Issue, 40(3), pp. 80–89

Sharp J. H., Ryan S. D. (2008). A Preliminary Conceptual Model for Exploring Global Agile Teams, P. Abrahamsson et al. (Eds.). *XP 2008*, LNBIP 9, pp. 147–160

Siakas K. V., Balstrup B. (2006). Software Outsourcing Quality Achieved by Global Virtual Collaboration, *Software Process: Improvement and Practice (SPIP) Journal*, John Wiley & Sons, Vol. 11, no. 3 (May-June)

Siakas K., Siakas E. (2008). The need for trust relationships to enable successful virtual team collaboration and software outsourcing, *The International Journal of Technology, Policy and Management, special issue on Human Aspects of Information Technology Development, Inderscience Enterprises*, Vol. 8, No.1 pp.59 - 75

Siakas K., Siakas E.(2007). The Agile Professional Culture: A Source of Agile Quality, *Software Process: Improvement and Practice (SPIP) Journal*, John Wiley & Sons, Volume 12, Issue 6, pp. 597 – 610

Siakas K. V., Siakas E. (2006a). Outsourcing: Achieving High Quality Software, in R. Dawson, E. Georgiadou, P. Linecar, M. Ross, G. Staples (eds), Perspectives in Software Quality, Proceeding of the *14th Software Quality Management Conference (SQM*), April, Southampton, UK, The British Computer Society, pp. 143-156

Siakas K., Siakas E. (2006b). The Human Factor Deployment for Improved Agile Quality, in Markku Tukianen, Richard Messnards, Risto Nevalaninen, Sonja Koining (eds), *European Software Process Improvement and Innovation (EuroSPI 2006*), International Proceedings Series 6, 11-13 October, Joensuu, Finland, Publisher: University of Joensuu, pp. 4.11- 23

Siakas K. V., Hyvärinen J.(2006). On-line Assessment of the Fit between National and Organisational Culture; A new tool for Predicting Suitable Software Quality Management System, in R. Dawson, E. Georgiadou, P. Linecar, M. Ross. G. Staples (eds), Perspectives in Software Quality, Proceeding of the *14th Software Quality Management Conference (SQM),* April, Southampton, UK, The British Computer Society, pp. 197-204

Siakas K. V., Balstrup B., Georgidaou E., Berki E. (2005). Global Software Development; the Dimension of Culture, *IADIS (International Association for Development of the Information Society) International Virtual Multi Conference on Computer Science and Information Systems (MCCSIS 2005) - SEA (Software Engineering and Applications),* 26 April 2005

Siakas K. V., Berki E., Georgiadou E. (2003). CODE for SQM: A Model for Cultural and Organisational Diversity Evaluation, *EuroSPI 2003 (European Software Process Improvement Conference*), Graz, Austria, 10-12.12.2003, pp. IX 1-11

Tippins M. J., Sohi R. S. (2003). It competency and firm performance: Is organizational learning a missing link? *Strategic Management Journal*, 24, pp. 745–761

Ware, L. C. (2003). Weighing the Benefits of Offshore Outsourcing, *CIO Research Reports*, retrieved at 23.03.2006 from http://www2.cio.com/research/surveyreport.cfm?ID=62

A CMMI Ontology for an Ontology-Based Software Process Assessment Tool

Sema Gazel, Ayça Tarhan, Ebru Sezer

Abstract

Process oriented management demands for time, effort, and qualified personnel from a software organization, and brings together some costs and risks in application. In case the organization is working with more than one process reference model and/or standard, tracking compliance of the organization's processes to each is getting more complicated. Many times, it becomes a necessity to utilize support tools to perform process management activities. Developing ontology of each process reference model and/or standard and representing it by using a shareable, machine-understandable language would be a suitable solution to provide such support. Therefore, with the purpose of developing an ontology based software process assessment tool, we developed the ontology of CMMI for Development by using Web Ontology Language and we have developed an Eclipse plug-in to display that ontology in Eclipse Process Framework. This paper presents in detail the ontology of CMMI for Development.

Keywords

CMMI, ontology, process model, process assessment tool

1 Introduction

The quality of a software product is highly influenced by the quality of the process used to develop and maintain it [1]. Various works have been demonstrated for modeling, assessing and improving software processes during the last decades. Some of them include a meta-model for software and systems process engineering called SPEM [2]; a framework for the assessment of processes called ISO/IEC 15504 [3]; and a process reference model that addresses development and maintenance activities for both products and services called CMMI [4].

Process improvement is a continuous activity as carrying out the Deming's plan-do-check-act steps on an organization's process assets in a cyclic way [5]. Definitions of process entities, setting the relationship between them, updating these definitions and relations when something is changed, and keeping the process assets under configuration control during process improvement, are important activities for organizations which adopted process oriented management [6]. Many times, it is almost a necessity to utilize support tools to perform process management activities.

Process assessment is the foundation step for process improvement activities. It investigates in detail strong, weak, and/or missing points in definition and application of a process [3]. Process assessment provides an understanding about current process situation and enables rating about process quality based on this understanding. It can be performed by *system/software engineering process group* of the organization, by independent consultants, or a combination of both [3][4]. Findings from a process assessment are usually transformed into issues for process improvement. Process assessments, either performed internally or externally to the organization, are very time and effort consuming and require qualified personnel. Especially, if an organization tries to keep its organizational processes in accordance with more than one process reference model and/or standard; tracking compliance of the organization's processes to each model and/or standard and finding out the deviation between them would be getting more complicated. When it is done manually, these activities can be error-prone, since the environment is open to mistakes and difficult to manage. If process modeling tools could be extended with some abilities to carry out process assessment activities, process assessments would be supported and it would be beneficial to reduce assessment process costs and risks.

In order to provide such an infrastructure; it is required to formally represent process reference models and/or standards so as to communicate with process management tools, to map the organization's process assets to these representations, and to inquire these mappings for strengths and weaknesses. Developing ontology of each process reference model and/or standard and representing it by using a shareable, machine-understandable language would be a suitable solution for creating such an infrastructure. Therefore, we first developed the ontology of CMMI-Dev [4], which is a widely accepted process reference model by the organizations developing system and software [1], and represented it in OWL (Web Ontology Language) [7] by using Protégé-OWL editor [8]. Secondly, we have developed an Eclipse [9] plug-in to display that ontology in Eclipse Process Framework (EPF) [10] which uses SPEM. As a current work, we aim to model the software development process of Aselsan Inc., the biggest defense industry corporation of Turkey developing systems and software, by utilizing EPF, to map these definitions to the CMMI-Dev ontology, and to inquire these mappings to derive assessment findings. This paper presents the output from the first step, which is the ontology of CMMI-Dev, in detail.

2 Background and Related Works

2.1 Capability Maturity Model Integration (CMMI)

CMMI is a process reference model that addresses the development and maintenance activities applied to both products and services. This model could be used for improving processes, and measuring the capability of a process or the maturity of an organization [4]. CMMI components (including a model, its training materials, and appraisal-related documents) are designed to meet the needs of some specific areas of interest, which is called *constellation*. There are three constellations as supported by the version 1.2 of the framework: *Development [4]*, *services [11]*, and *acquisition [12]*. CMMI for Development (CMMI-Dev) is one of these constellations.

CMMI consists of process areas, goals and practices of these process areas, two different representations, and two different scopes. Representations are *staged* and *continuous*, and scopes are *with IPPD* (Integrated Product and Process Development) and *without IPPD*. The representations and the scopes indicate in what way the goals and practices shall be handled. The representations can be considered as two different viewpoints created by putting the model components together in two different ways. IPPD, which is an addition, enables expansion of the process areas in CMMI with goals and practices so as to cover the integrated team activities.

Process Area: A cluster of related practices in an area that, when implemented collectively, satisfy a set of goals considered important for making improvement in that area. In CMMI-Dev, all software and system development processes are handled in twenty two process areas, such as Requirements Management, Project Planning, etc. In CMMI, all process areas have the same components and component structure is illustrated in Figure 1.

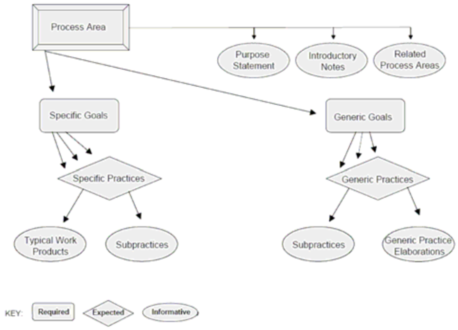


Figure 1 – CMMI Model Components [4]

Goals comprise of a goal statement and a number of *practices*. Goal statement describes the goal to be satisfied, and practices address the steps which enable the goal's satisfaction. In process assessments, satisfaction of goals is a necessity. Practices, on the other hand, can be implemented by an organization as different from the suggestions of the model. In other words, practices could be achieved in a different way in the organization as long as they serve the goals to be reached.

There are two types of goals in the model; *generic goals (GGs)* and *specific goals (SGs)*. SGs belong to a specific process area and enable an implementation of that process area. Practices of a SG are called *specific practices (SPs)*. SGs are different for each process area in terms of content and the number of goals and practices involved. GGs of a process area, on the other hand, serve institutionalization of that process area. Practices of a GG are called *generic practices (GPs)*. Since institutionalization is similar for all process areas, the content and the number of goals and practices involved are repeated for each. There are 5 GGs in each process area: GG1, GG2, GG3, GG4, and GG5.

Representations: CMMI involves two different representations called staged and continuous. They enable an organization to handle model components and to be assessed using two different viewpoints. In continuous representation, process areas can be handled individually for improvement. In staged representation, the process areas are handled in groups as defined in the model.

Capability and Maturity Levels: There are six capability levels, numbered from 0 to 5; and five maturity levels, numbered from 1 to 5. 5 means the most improved in both ratings. In CMMI, each level constitutes a basis for the next level. So, whether it is a maturity rating or a capability rating, in order to be considered successful in a level, it is required that the previous levels should be covered as well as those that are required to be covered at this level.

2.2 Related Works

Ontologies are content theories about the sorts of objects, properties of objects, and relations between objects that are possible in a specified domain of knowledge [13]. They provide potential terms for describing our knowledge about a domain. Ontological analysis clarifies the structure of knowledge and ontologies enable knowledge sharing in a selected domain [13]. The knowledge in a domain can be shared with others who have similar needs for knowledge in that domain, thereby eliminating the need for replicating the knowledge-analysis process. The ontology represented with shareable language can thus form the basis for domain-specific knowledge-representation [13].

A *software process model* is an abstract representation of the architecture, design or definition of the software process [14]. Liao et al. present some problems and difficulties in the usage of the process models and how ontologies can eliminate problems and make easier the difficulities [15]: **1)** Formal description of process models: Process models lack rigorous and formal description of model structure and process framework. Problems of ambiguity, instability, too much subjectivity, and inaccuracy in process assessment and application were identified in existing process models. Ontology can eliminate conceptual and terminological confusion, and provides a representation vocabulary specialized to the software processes. **2)** Compatibility and transformability: By creating ontologies for the current process models and using ontology alignment techniques, the compatibility problem can be solved without the cost of changing the existing models. **3)** Benchmark of process attributes: With ontology and semantic web, collecting data from the Internet and developing benchmarks of software processes in some areas would become easier.

There are only a few CMMI ontology works in literature. **Soydan et al.** presented OWL Ontology for CMMI-SW [16]. In this work, only staged representation was analyzed, whereas we aimed to develop a CMMI-Dev ontology which meets the needs of both representations, staged and continuous, of CMMI. **Sharifloo et al.** [17] introduced an ontology developed to represent the CMMI-ACQ constellation which is another CMMI constellation than we aimed. This ontology was based on SUMO [18] upper ontology using SOU-KIF [19] languages. **Lee et al.** [20] presented an ontology-based computational intelligent multi-agent system for CMMI assessment. The system could summarize evaluation reports with using three agents. In that study, quality assurance ontology was built based on PPQA (Process and Product Quality Assurance) process area of CMMI. In another **Lee et al.** [21] study, they presented an ontology-based intelligent decision support agent (OIDSA) to apply to Project Monitoring and Control (PMC) process area of CMMI. The OIDSA was composed of three agents to find out the percentage of project progress completion for each project member in order to evaluate performance. In that study, Requirement Management, Project Planning and Project Monitoring and Control process areas of CMMI were considered for the ontology. In both Lee et al. studies, they focused some specific process areas as related to their purposes and represented the knowledge of those process areas; whereas we intended to cover all process areas in CMMI-Dev, though not representing the knowledge as specific to each process area but considering the structure of the CMMI, in our ontology. Furthermore, in these studies, [20] and [21], the ontologies were constructed according to a domain ontology structure which was also given in these studies. In other words, CMMI was used as an instance of the domain ontology. However, in our study, CMMI was selected as the domain itself, and ontology was constructed by considering the concepts and relations within the model. **Liao et al.** [15] aimed at creating generic software process ontology and strived to ensure that it covered the requirements of both CMMI and ISO/IEC 15504, whereas we targeted to develop ontology only for CMMI-Dev and to represent its domain knowledge as much as possible.

3 CMMI Ontology

This work has been initiated with the purpose of creating ontology of CMMI-Dev, mapping an organization's process assets to this ontology, and inquiring this mapping for strengths and weaknesses as related to a certain maturity and/or capability level. The work started with the selection of “continuous” as the representation and “without IPPD” as the scope; and at first, a CMMI ontology for this selection was developed. Second, a CMMI ontology for “staged” representation with the scope “without IPPD” was developed. Then, by considering the purpose of our work, common and distinct parts of the two

CMMI ontologies were assessed and it was concluded that the two ontologies could be expressed within a single CMMI ontology. While aggregating the ontologies, special care was devoted to preserve the correctness of concepts and their relations as specific to each representation, and to correctly combine the components that are common to both representations. It should also be noted that although we aggregated the two CMMI ontologies regarding “continuous” and “staged” representations for the scope “without IPPD” and developed the ontology of CMMI-Dev, this ontology can also be valid for other constellations (namely CMMI for Services and CMMI for Acquisition) as well. This is because version 1.2 of CMMI unifies the concepts and their relations for all constellations and it only distinguishes the domains of application, including process areas, specific goals, and specific practices, among the constellations. However, the validation of the ontology for other two constellations is beyond the scope of this work.

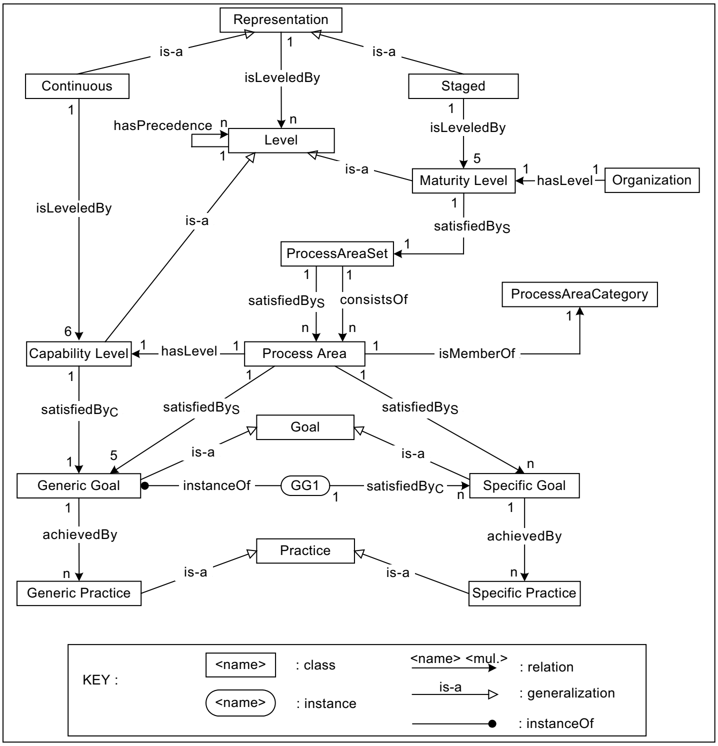


Figure 2 – The Aggregated CMMI Ontology

The aggregated CMMI Ontology regarding “continuous” and “staged” representations for the scope “without IPPD” is illustrated in Figure 2. As a result of the aggregation, an association between the two representations was established. This association enables changing the representation to be used and allows tracking of what a level in a representation corresponds to in the other. The ontology of CMMI-Dev that we developed by using OWL can be reached via the web address [22].

The relations in Figure 2 and their meanings are described below.

is-a relation: All is-a relations in the CMMI ontology demonstrate the superclass-subclass relationship between the classes.

hasLevel relation: An organization has a maturity level when its processes are assessed as a set of process areas at organizational basis. This is demonstrated by <hasLevel> relation between <Organi-

zation> and <Maturity Level> objects. A process area has a capability level when it is assessed at process area basis. This is shown by <hasLevel> relation between <Process Area> and <Capability Level> objects.

isLeveledBy relation: Staged representation is rated by maturity levels, and this is demonstrated by <isLeveledBy> relation between <Staged> and <Maturity Level> objects. Continuous representation is rated by capability levels, and this is shown by <isLeveledBy> relation between <Continuous> and <Capability Level> objects.

hasPrecedence relation: A level's achievement requires the satisfaction of all previous levels and this is demonstrated by <hasPrecedence> relation on the <Level> object. This relation was previously shown on <Capability Level> and <Maturity Level> objects in the CMMI ontologies of continuous and staged representations respectively; but was moved onto <Level> object which is the ancestor class of both <Capability Level> and <Maturity Level> objects in the aggregated CMMI ontology.

consistsOf relation: All process areas in CMMI-Dev are divided into four process area sets, each of which is associated with a maturity level. Each process area is a member of one and only one of these sets. This is demonstrated by <consistsOf> relation between <ProcessAreaSet> and <ProcessArea> objects.

isMemberOf relation: In CMMI, process areas are organized into some categories, which is four in CMMI-Dev, and each process area is a member of one and only one of these categories. This is represented by <isMemberOf> relation between <Process Area> and <Process Area Category> objects.

satisfiedBy$_S$ relation (towards <ProcessAreaSet> object): For an organization to have a maturity level, the process area set associated to related maturity level should be satisfied. This is demonstrated by <satisfiedBy$_S$> relation between <Maturity Level> and <ProcessAreaSet> objects. The subscript letter "S" stands for "staged" to show that this relation is valid only for the staged representation.

satisfiedBy$_S$ relation (towards <ProcessArea> object): For satisfying a process area set, all process areas in that set should be satisfied. This is demonstrated by <satisfiedBy$_S$> relation between <ProcessAreaSet> and <ProcessArea> objects.

satisfiedBy$_S$ and satisfiedBy$_C$ relations (towards Goals): Goals are required components and they must be satisfied. This is demonstrated by *satisfiedBy* relations in the CMMI ontology. However, there is a difference in this relation in the initiating objects for staged and continuous representations, and subscript letters 'C' and 'S' are attached to the name of the relation to indicate this difference. In the staged representation, a process area is satisfied by the achievement of its GGs and SGs; therefore the relation is named as <satisfiedBy$_S$> and is initiated from <Process Area> object towards <Generic Goal> and <Specific Goal> objects. In the continuous representation, a capability level of a process area is satisfied by the achievement of its GG; therefore, the relation is named as <satisfiedBy$_C$> and is initiated from <Capability Level> object towards <Generic Goal> object. Another difference specific to the continuous representation is that GG1 is satisfied by the achievement of SGs of a process area. This is shown by <satisfiedBy$_C$> relation between <GG1> instance of <Generic Goal> object and <Specific Goal> object.

achievedBy relation: A goal is satisfied if all its practices are achieved, and this is valid for both GGs and SGs. This is demonstrated by <achievedBy> relation between <Generic Goal> and <Generic Practice> objects, and between <Specific Goal> and <Specific Practice> objects.

instanceOf relation: This relation takes place between a class and an instance of it.

4 Conclusions and Current Works

If organizations are able to map each process reference model and/or standard, which they intend to be in compliance with their organizational process definitions, by using tools, and update and inquire these mappings when required, they will obtain some advantages. Any deviations from the models and/or standards could be seen instantaneously and thus the process assessment will be easier than doing it manually. Besides, any deviations from the models and/or standards could be used as input for process improvement activities at other times than the process assessments.

From that view point, we have aimed to create an ontology based software process assessment tool, and as the first part, we developed the ontology of CMMI for Development by using Web Ontology Language as well as an Eclipse plug-in to display that ontology in Eclipse Process Framework. CMMI-Dev ontology and its role in the tool are presented here. Currently, we have been working on the realization of the tool, and results will be announced.

As a current work, modeling of software development process is being performed by using EPF for an organization, which is Aselsan Inc., the biggest defense industry corporation of Turkey developing systems and software. Aselsan Inc. intends to improve its processes based on CMMI-Dev, and to have an assessment in accordance to maturity level 3. We should note that this is why we chose CMMI-Dev as basis for creating the ontology. Creating ontology of other reference models such as ISO/IEC 12207 [23] and/or ISO IEC 15288 [24] could serve the same target for organizations.

As the next step, the plug-in will be extended to map the organization’s process definitions to the goals and practices in the ontology of CMMI-Dev. Then, by using EPF, some queries will be performed on this mapping so as to have information about the conformity of the organization’s process definitions to targeted maturity and/or capability levels. We may query, for example, existing applications or work products of a process matching with a certain maturity and/or capability level. Another example could be querying CMMI goals and practices which can not be mapped to a process asset and/or to an application in the organization, mostly because of the lack of process practice.

We believe that the CMMI-Dev ontology explained in this paper will serve both towards easy and accurate understanding of CMMI, and towards other similar studies.

5 Literature

[1] Humprey. A Discipline for Software Engineering. Addison-Wesley, 1995.

[2] Software & Systems Process Engineering Metamodel Specification (SPEM), version 2.0, OMG Document Number: formal/2008-04-01, Object Management Group (OMG), 2008.

[3] ISO, ISO/IEC 15504: Information Technology - Process Assessment, 2003-2008.

[4] CMMI Product Team, CMU/SEI. CMMI for Development. CMMI-SE/SW V1.2, CMU/SEI-2006-TR-008, 2006.

[5] N.R. Tague. The Quality Toolbox, 2nd Ed., ASQ Quality Press, pp. 390-392, ISBN 978-0-87389-639-9, 2004.

[6] ISO, ISO 9001:2000 Quality Management Systems -- Requirement, 2000.

[7] W3C. Web Ontology Language Reference OWL, 2004. http://www.w3.org/2004/OWL/

[8] Stanford Center for Biomedical Informatics Research, Protégé-OWL Ontology Editor, version : 3.3.1. http://protege.stanford.edu/overview/protege-owl.html

[9] Eclipse Foundation. http://www.eclipse.org/

[10] Eclipse Process Framework Project (EPF). http://www.eclipse.org/epf/

[11] CMMI Product Team, CMU/SEI. CMMI for Services. CMMI-SVC V1.2, CMU/SEI-2009-TR-001, February 2009.

[12] CMMI Product Team, CMU/SEI. CMMI for Acquisition. CMMI-ACQ V1.2, CMU/SEI-2007-TR-017, November 2007.

[13] B. Chandrasekaran, J.R. Josephson, V.R. Benjamins. What Are Ontologies, and Why Do We Need Them? IEEE Intelligent Systems Vol. 14, Issue 1, pages 20-26, 1999.

[14] P.H. Feiler, W.S. Humphrey. Software Process Development and Enactment: Concepts and Definitions. Proceedings of the Second International Conference on Software Process, February 1993.

[15] L. Liao, Y. Qu, H. Leung. A Software Process Ontology and Its Application. In ISWC 2005 Workshop on Semantic Web Enabled Software Engineering, 2005.

[16] G.H. Soydan, M.M. Kokar. An OWL Ontology for Representing the CMMI-SW Model. In ISWC 2006 Workshop on Semantic Web Enabled Software Engineering, 2006.

[17] A.A. Sharifloo, Y. Motazedi, M. Shamsfard, R. Dehkharghani. An Ontology for CMMI-ACQ Model. In 3rd International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA), April 2008.

[18] Suggested Upper Merged Ontology (SUMO). http://www.ontologyportal.org/

[19] Knowledge Interchange Format, draft proposed American National Standard (dpANS), NCITS.T2/98-004, Ed.

[20] C.-S. Lee, M.-H. Wang. Ontology-based Computational Intelligent Multi-agent and Its Application to CMMI Assessment. Springer Science+Business Media, LLC 2007.

[21] C.-S. Lee, M.-H. Wang, J.-J. Chen, C.-Y. Hsu. Ontology-based Intelligent Decision Support Agent for CMMI Project Monitoring and Control. International Journal of Approximate Reasoning, volume 48, pages : 62–76, April 2008.

[22] The CMMI-Dev Ontology. http://web.cs.hacettepe.edu.tr/~atarhan/cmmi-owl/

[23] ISO, ISO/IEC 12207: Systems and software eng. - Software life cycle processes, 2008.

[24] ISO, ISO/IEC 15288: Systems and software eng. - System life cycle processes, 2008.

6 Author CVs

Sema Gazel

Sema Gazel received her B.Sc. degree in Computer Engineering from Hacettepe University, Ankara, Turkey, in 1997. She has been working for Aselsan Inc. since then. In Aselsan, she worked as a software developer and for the last two years, she has been working on process improvement based on CMMI. She is also currently working to get her M.Sc. degree in Computer Engineering at Hacettepe University. Her research interests include ontology and CMMI. Contact her at Aselsan Inc. PO Box 1, 06172 Ankara/Turkey. E-mail: gazel@aselsan.com.tr or semagazel@gmail.com

Ayça Tarhan

Ayca Tarhan works as consultant and researcher in the area of software engineering for ten years. She has a strong experience on model-based assessment and improvement of software processes and pursues her studies with a focus on software measurement, process management, and software engineering standards. She has a Ph.D. in Information Systems and a M.Sc. in Computer Engineering. Contact her at Computer Engineering Department of Hacettepe University, 06532, Ankara/Turkey. E-mail: atarhan@hacettepe.edu.tr

Ebru Sezer

Ebru Sezer graduated from Computer Engineering Department of Hacettepe University (HU) at 1996. She completed her master thesis on Information Retrieval at 1999 and doctorate thesis on Semantic Web at 2006 in the same department. She has been working as a Lecturer (Ph.D.) at Computer Engineering Department of HU since 2006. Contact her at Computer Engineering Department of Hacettepe University, 06532, Ankara/Turkey. E-mail: esezer@cs.hacettepe.edu.tr

Utilizing a Metrics Knowledge Base for Software Process Improvement

Timo Mäkinen, Jari Soini and Timo Varkoi
Tampere University of Technology,
Pohjoisranta 11A, FI-28100 Pori, Finland
{timo.makinen, jari.o.soini, timo.varkoi}@tut.fi

Abstract. We present an approach to provide specific information to support process improvements. In this case we use an existing software engineering metrics knowledge base and integrate it with a comprehensive method for process assessments and modeling. The goal is to effectively assist practical process improvement by linking measurement information with the actual process elements. The method under development produces proscriptive process models that contain detailed information to be applied in process improvement. In the future the method will be extended with additional knowledge bases and its efficient use will be supported by tools.

Keywords: software process, process assessment, process modeling, measurement

1 Introduction

This paper presents an approach for using an existing knowledge base that contains categorized software engineering metrics to support process improvement and innovation. We also present the underlying assessment and modeling method and the knowledge base for software engineering metrics integrated with the method.

The assessment and modeling framework, Assessment Driven Process Modeling (ADPM) [1],[2], is a comprehensive methodology that is under development, which combines process assessment and modeling to create descriptive process models. ADPM also supports the development of target process models based on target process capabilities. The current development phase integrates external process knowledge bases so that specific improvements can be described with supporting instances of their implementation. Here we use software process metrics as an example to illustrate the possibilities.

The software metrics database in this case is the result of a research project that collected the metrics that are actually used in the software industry. In addition, the metrics were classified using existing process reference models. This invaluable information has now been utilized to support detailed process improvements.

The implementation of the improvements can be supported by providing practical examples of e.g. measures related to the required practices and work products. A specific metrics example contains substantially more practical information for the assessed organization than just a process capability rating with improvement targets.

In the next section, we will briefly introduce the three phases of ADPM: *Elicitation*, *Targeting,* and *Resolution*. The third section discusses the metrics knowledge base that provides, as an example, the process measurement related information to support process improvements. In the fourth section there is a simple example of the process models that ADPM produces, including a scenario of how a metrics knowledge base can be utilized along with ADPM. Our ideas for future development are summarized in section five.

2 Assessment Driven Software Process Modeling

The foundation for this work is our decade-long research co-operation with the software industry and our experience with the numerous software process improvement (SPI) programs in software-producing organizations. One of the findings is that specific improvements are often tedious to implement, because the organization using the process assessment results may not see a direct link with or effect on their operations. Typically, process assessment or modeling describes the present situation of the studied process. Missing process elements can be spotted using the information in the process models or by comparing the process against a process meta-model. Based on the organization's business needs, a target process capability can be defined. The gap between the present and target capability shows the required process improvements. Our goal is to concretize the improvements by giving practical examples of solutions that have been applied in similar situations.

Process assessment is a means of characterizing the status of a software process for improvement purposes. The main result of an assessment is a set of Process Profiles, which describes the difference between the assessed processes and reference processes. Some Target Process Profiles are proposed on the basis of the assessment results and the organization's business goals. Usually, the output of an assessment also includes textual Suggestions of how the target profiles can be implemented in practice [3].

Assessment Driven Process Modeling (ADPM) [1],[2] is an approach that produces modeling counterparts for the results of an assessment: a *Descriptive Process Model*, a *Generic Target Process Model*, and *Prospective Process Models*. A *Process Library* and *Knowledge Base* support the construction of process models [4],[5]. ADPM also suggests storing its results in the organization's *Process Assets Library* [6]. Fig. 1 depicts the context, phases, and main outcomes of process assessment and ADPM. In the figure, the Metrics Knowledge Base is emphasized as part of the Process Knowledge Base.

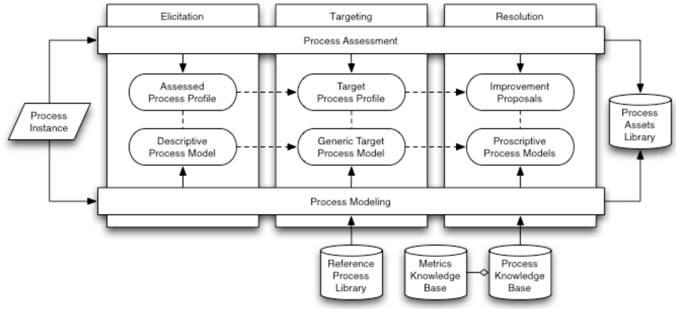


Fig. 1. The context, main phases, and outcomes of process assessment and Assessment Driven Software Process Modeling (ADPM).

Assessment utilizes the elements, known as indicators, of an assessment model to classify the information related to processes, which are rated in terms of the indicators. In the *elicitation phase*, ADPM first associates the indicators with the process elements of the organization, and then discovers related process elements guided by a process meta-model. Next, the indicator-based process models are integrated into the resulting descriptive process model.

In the *targeting phase*, ADPM adds elements from the reference processes (assessment indicators) into the descriptive process model to fill the gap between the target and actual process profiles. The result is a generic target process model, which is the counterpart of a target process profile.

Assessment models with their indicators do not describe complete software development methods. They merely include checklists indicating good software practice without any guidance on how to implement what the checklist items require. The *resolution phase* of ADPM transforms the generic target process model into prospective process models by replacing the elements from reference processes with elements from practical software engineering and management methods. The prospective process models are the counterparts of textual proposals for improvements, which are frequently included in the result of an assessment.

3 Example of a Metrics Knowledge Base

This section presents one example of the elements comprising a process knowledge base, i.e. the metrics knowledge base created during the completed SoMe (Software Measurement) research project. This section describes briefly how the knowledge base was generated, and presents the final outcome of the project - an information system implemented in a web environment based on a large metrics knowledge base. This developed information system is meant for individuals and organizations seeking appropriate software metrics and measurement practices for their needs.

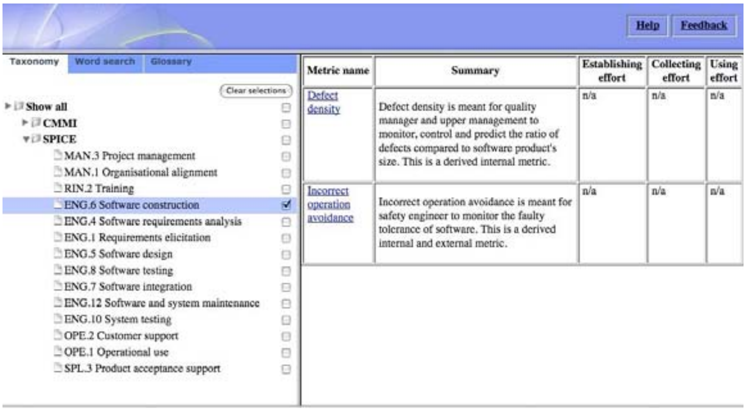
The SoMe research project was carried out between 2005 and 2007, coordinated by Tampere University of Technology (TUT) [7]. The goal of the research project was to generate a metrics knowledge base and to develop a support system to enhance measurement knowledge and the use of these metrics, and also to facilitate the availability and distribution of metrics information. The implemented information system was released together with its web-based application for measurement knowledge transition at the end of the 20-month research project in April 2007. The population of the study presented in this thesis consists of Finnish software companies all over Finland, which are members of FiSMA (the Finnish Software Measurement Association) [8]. Additionally, the sample of FiSMA companies was not made randomly, using the post-trial adjustment [9] approach instead. Purposive sampling [10] was used in this study, which means in this case that the choice of case companies was based on the rationale that new knowledge related to the research phenomenon would be obtained. Other publications in relation to the SoMe project [11],[12] describe in more detail the process used for capturing, modifying, evaluating and distributing the measurement knowledge via the information system that was developed.

The research was an empirical study, based on interviews and a questionnaire format. The knowledge required has been gathered from the participating organizations through externalization, using face-to-face meetings and a data collection form as the mechanism. The same structured interview templates were used in all interview sessions: one form to collect general information about the company and its measurement practices, and another form to collect all the metrics the company uses or has used. When planning the spreadsheet form, a qualitative research perspective was selected. The aim was to gain as explicit and in-depth a description as possible of all the software engineering process and product metrics used in the participating companies. This is related to the goal of the SoMe project, which was to describe the individual metrics information in the database at such a detailed level that it is possible to establish and adopt the metric according to the description.

For process improvement work, organizations need a deeper understanding of their own processes and to do this they need measurement data [13]. With this measurement information they can reliably seek and find improvement objects in their processes [14],[15],[16]. This approach was selected as the starting point of the development work and steered the work throughout system implementation. This aspect also guided the search taxonomy design of the information system. The selected search taxonomy was created based on the CMMI [17] and SPICE [18] process assessment models (see Fig. 2 below). The aim of this selection was for the organizations to familiarize themselves with and utilize these assessment models in their operations. Moreover, the information system also includes a word search option (see top left of Figure 2) to search for a suitable metric. This feature was included because there may be organizations that are not familiar enough with the process assessment models to start using a system based on them.

The information system developed works on a database that contains knowledge about software metrics and also measurement practices used in software organizations. The metrics knowledge base consists of individual items of information, *knowledge items* (individual metrics). A standard form, *a metric document*, is used for presenting each knowledge item. The formula for the title level and the terminology used in all metric documents is congruent. This solution helps the end user to read, perceive the logic, and make a comparison between the metrics.

In the information system, the *knowledge items* are linked to the process groups inside the assessment models. Every knowledge item also includes information for all the process groups to which it relates and in practice is linked to. The system offers a bi-directional link, from software processes to software metrics, and vice versa. This link between processes and metrics helps and advises the users to identify the relevant metrics for controlling a particular process. This characteristic also enables the user to see the dependence between process groups from a metric viewpoint and gives important information when planning measurement activities (e.g. a measurement program). These connections are also seen from the process group viewpoint, as the proper metrics depend on the selected process group in the selected assessment model (SPICE or CMMI). This realization method enhances awareness of the relationship between process assessment and process measurement. As an example, Fig. 2 (below) presents the results of one search: SPICE assessment model / (Engineering process group) ENG 6 Software Construction.



- Show all □
  - CMMI □
  - SPICE □
    - MAN.3 Project management □
    - MAN.1 Organisational alignment □
    - RIN.2 Training □
    - ENG.6 Software construction ☑
    - ENG.4 Software requirements analysis □
    - ENG.1 Requirements elicitation □
    - ENG.5 Software design □
    - ENG.8 Software testing □
    - ENG.7 Software integration □
    - ENG.12 Software and system maintenance □
    - ENG.10 System testing □
    - OPE.2 Customer support □
    - OPE.1 Operational use □
    - SPL.3 Product acceptance support □

Metric name	Summary	Establishing effort	Collecting effort	Using effort
Defect density	Defect density is meant for quality manager and upper management to monitor, control and predict the ratio of defects compared to software product's size. This is a derived internal metric.	n/a	n/a	n/a
Incorrect operation avoidance	Incorrect operation avoidance is meant for safety engineer to monitor the faulty tolerance of software. This is a derived internal and external metric.	n/a	n/a	n/a

Fig. 2: User interface (UI) of the web-based information system.

After making the selection, the results (individual metrics) of this search appear in the information system display (see right of Figure 2). The user can see directly all the related metrics and also a brief description such as: the name of the metric(s), a short summary of each metric, and workload evaluation for establishing, collecting, and using the metric. Depending on the given search selection, the system retrieves the particular metrics from the knowledge base that are linked to the selection (individual process group inside the assessment model). Selecting a particular metric (by clicking on the metric name field) calls up the detailed information on this metric.

Process measurement is regarded as the key means to obtain feedback to drive the improvement effort [19],[20],[21],[22],[23]. There seems to be a connection between software process improvement and software process measurement, but the link is not explicit. The system developed also advises the user on how to use measurement as an instrument for software process improvement. This metrics knowledge base is just one example of the elements comprising a process knowledge base (see Fig 1). In the next section there is a discussion of how to integrate this example to the assessment and modeling framework (ADPM), which is the focus of this paper.

4 Scenario for Utilizing a Knowledge Base

In this section we depict a scenario of how the presented SoMe metrics knowledge base can be utilized along with ADPM. Fig. 3 presents a fictitious example of the elicitation of a descriptive process model. In the example, ISO/IEC 15504-5 [18] is employed as the assessment model and its base practices are utilized as assessment indicators. The Software Construction process (ENG.6) is in the scope of the assessment, and therefore its base practices are mapped to the tasks of the assessed organizational unit. The process meta-model of the example is extremely simple, containing only one task and one artifact, and the relationships between them.

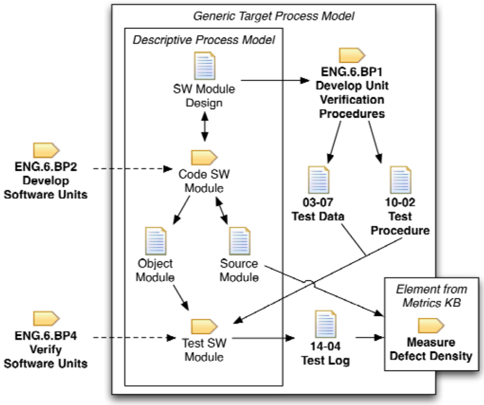


Fig. 3. The resulting process models of the ADPM, and their relationships.

To produce the *descriptive process model*, the base practice, ENG.6.BP2 Develop software units, is mapped to the organizational unit task, Code Module. Then the process elements related to the task are identified. Three artifacts (Module Design, Source Module, and Object Module) are found to be associated with the Code Module task. The base practice, ENG.6.BP4 Verify software units, is treated similarly.

The descriptive process model depicted in Fig. 3 does not fully reach the requirements of the base practices of the ENG.6 process. To locate the subject to improve in the organizational unit process, adding four process elements into the descriptive process model produces a *generic target process* model. The additional elements (ENG.6.BP1 Develop Unit Verification Procedures, 03-07 Test Data, 10-02 Test Procedure, and 14-04 Test Log) are the assessment indicators of ENG.6. Prospective process models can be constructed by replacing the generic elements with elements of a practical software engineering method, such as the OpenUP [2].

The implementation of the improvements can be supported by providing practical examples of e.g. measures related to the required practices and work products. In this example a measure, Measure Defect Density, can be found from an external process knowledge base based on the classification. A specific metrics example contains substantially more practical information for the assessed organization than just a process capability rating with improvement targets.

5 Conclusions

This paper presents an approach for integrating various knowledge bases to provide supportive information for software process improvement. The starting point is a comprehensive method, ADPM, which enables the use of multiple sources of information to generate detailed proposals for process improvement.

In our example, a metrics knowledge base was used to supplement the process models that were created based on an assessment. Measurement is vital in pursuit of higher process capability. For instance, SPICE [18] Capability Level 4 has a process measurement attribute that requires an organization to 'identify product and process measures that support the achievement of the quantitative objectives for process performance'.

The benefits of this approach include precise and concrete improvements to the software process, as well as fast and accurate elicitation of the processes' current state. On the other hand, the method relies heavily on expert judgment – the contents and applicability of the knowledge used and its mapping to the processes is, of course, debatable.

Currently the method is not yet complete; the higher capability levels of process improvement have not yet been addressed. However, the method is a starting point in providing useful improvement guidance, based on the assessed situation of the software process.

In the future, ADPM will be extended to utilize other sources of experiences of software process improvement already collected and analyzed. Furthermore, the method will address the needs related to higher organizational maturity, i.e. higher capability levels of the selected processes. Efficient use of the methodology requires supporting tools that are self-explanatory even when used by small software-producing entities.

References

1. Mäkinen, T., Varkoi, T. (2008a), Assessment driven process modeling for software process improvement, in Proceedings of PICMET'08 Conference (Portland International Center for Management of Engineering and Technology), 27-31 July 2008, Cape Town, South Africa.
2. Mäkinen, T., Varkoi, T. (2009), A three-phase Approach for Assessment Driven Process Modeling, unpublished.
3. Mäkinen, T., Varkoi, T., Lepasaar, M. (2000), A detailed process assessment method for software SMEs, in Proceedings of the EuroSPI 2000 Conference, November 7-9, 2000, Copenhagen, Denmark. pp. 1.14–1.26.
4. Alexandre, S., Mäkinen, T., Varkoi, T. (2008), Implementation of a software process standard as an electronic process guide, in Proceedings of SPICE 2008 Conference (Software Process Improvement and Capability dEtermination), 26-28 May 2008, Nuremberg, Germany.
5. Mäkinen, T., Varkoi, T. (2008c), Software process assessment model as a reusable process library, in Short Papers Proceedings of Profes 2008 Conference (Product Focused Software Process Improvement), 23 - 25 June 2008, Frascati, Italy.
6. Mäkinen, T. & Varkoi, T. (2008b), Managing assessment and modeling results as process assets, in Short Papers Proceedings of Profes 2008 Conference (Product Focused Software Process Improvement), 23 - 25 June 2008, Frascati, Italy.
7. TUT, Tampere University of Technology, (2009), http://www.tut.fi/
8. FiSMA, Finnish Software Measurement Association, (2009), http://www.fisma.fi/eng/index.htm.
9. Kitchenham, B., Pfleeger, S. L., Pickard, L., Jones, P., Hoaglin, D., El Emam, K., Rosenberg, J. (2002), Preliminary Guidelines for Empirical Research in Software Engineering, IEEE Transactions on Software Engineering, vol. 28, no. 8, pp. 721-733.
10. Curtis, S., Gesler, W., Smith, G., Washburn, S. (2000), Approaches to sampling and case selection in qualitative research: Examples in geography of health, Social Sience & Medicine, vol. 50, issue 7-8, pp. 1001-1014.
11. Soini, J., Varkoi, T., Tenhunen, V., Tukiainen, M. (2007a), Empirical case study of software metrics using the SPICE framework, in proceedings of the International SPICE Conference (SPICE'07), Seoul, Korea, May 9-11
12. Soini, J., Tenhunen V., Mäkinen, T. (2007b), Managing and Processing Knowledge Transfer between Software Organizations: A Case Study, in proceedings of the International Conference on Management of Engineering and Technology (PICMET'07) Conference, pp. 1108-1113.
13. Briand, L., Differding, C., Rombach, D. (1997), Practical Guidelines for Measurement-Based Process Improvement. Software Process Improvement and Practice, vol. 2, no. 4, pp. 253-280.
14. Dybå, T. (2000), An Instrument for Measuring the Key Factors of Success in Software Process Improvement. Empirical Software Engineering, vol. 5, no. 4. pp. 357-390.
15. Pfleeger, S. L., Rombach, H. D. (1994), Measurement Based Process Improvement, IEEE Software, July, pp. 9-11.
16. Rodenbach, E., van Latum, F., van Solingen, R. (2000), SPI – A Guarantee for Success? – A Reality Story from Industry, in proceedings of the PROFES 2000 Conference, (eds.) Bomarius, F. and Oivo, M., Oulu, Finland, pp. 216-231.
17. CMMI for Systems Engineering/Software Engineering, Version 1.02 (2000), CMU/SEI-2000-TR-029. Carnegie Mellon University, SEI.

18. ISO/IEC (2006). Information technology – process assessment – part 5: An exemplar process assessment model. International Standard ISO/IEC 15504-5:2006, International Organization for Standardization, Switzerland, 2006
19. Dybå, T. (2003), 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 (ESEC´03), pp. 148-157.
20. Dybå, T. (2005), An Empirical Investigation of the Key Factors for Success in Software Process Improvement, IEEE Transactions on Software Engineering, vol. 31, no. 5, pp. 410-424.
21. Hall, T., Baddoo, N., Wilson, D. N. (2000), Measurement in Software Process Improvement Programmes: An Empirical Study, in proceedings of the 10th International Workshop on Software Measurement (IWSM 2000), pp. 73-83.
22. Parnas, D. L., Clemens, P. C. (1986), A Rational Design Process: How and Why to Fake It, IEEE Transactions on Software Engineering, vol. SE-12, no. 2, pp.251-257.
23. Rainer, A., Hall, T. (2003), A quantitative and qualitative analysis of factors affecting software processes, Journal of Systems and Software, vol. 66, no. 1, pp. 7-21.

Authors

Timo Mäkinen (Lic. Tech.) works at the Tampere University of Technology (TUT) as a senior researcher. He has eight years' practical experience in software engineering as a programmer, systems analyst, and project manager. He has been lecturing in data management and software engineering for over ten years. He has published more than forty articles related to software process improvement (SPI). His current research interests include software process modeling and knowledge management in SPI.

Jari Soini (Dr. Tech.) works at the Tampere University of Technology (TUT), Pori, Finland, as a researcher and research project manager. He has four years' experience in the software industry in implementing software products, software consulting, and managing software projects. Currently his duties at TUT include managing software research projects and acting as an expert in software measurement related issues in education, as well as collaboration with the software industry. His current research interests include software process modeling, measurement, and improvement-related research. He is also a member of the Centre of Software Expertise unit (CoSE) at the Tampere University of Technology, Pori.

Timo Varkoi (Lic. Tech.) is currently working as a research project manager at the Centre of Software Expertise (CoSE) at the Tampere University of Technology, Pori. His working experience includes both software development and management responsibilities in industrial software organizations. He is a competent SPICE assessor, a board member of the Finnish Software Measurement Association (FiSMA), and an active participant in ISO/IEC Software and Systems Engineering standardization. His current interests include software process assessment and improvement-related research, expertise, and training combined with the development of software-intensive organizations.

Software process improvement within a software R&D department

Mark Kevitt,
Renaat Verbruggen
Dublin City University

Why implement a Software Quality Improvement Process

The software industry is a volatile engineering discipline with substantial demands placed on teams for delivering excellence. Excellent products however come at a price, both financial and chronological; how can you create a superior software product which is intangible, complex and under ever reducing budgets and time pressures?

The answer is comparable to sailing; a winning sailor senses the wind, the sea, and fine tunes the sails, adjusts the boat balance and steers a good course with one eye on his competitors. A software product floats on a sea of people with one helm, tuning the team's efforts during each leg of development, and constantly steering the product through iterative development, navigating to the needs of the customer. Both sailing and software development are team activities with one goal and many challenges. In either activity you can sink or be plain sailing.

Superior software has a high standard of quality, achieving this standard requires a strategy; in software development, a software process is required. The process should consist of all team participants' active involvement throughout the product lifecycle. The process begins with end users and also terminates with end users. Each team participant must have an understanding of what is expected of them throughout the cycle with deadlines and measurable entry and exit criteria. The process which I have first hand at creating and implementing is based on implementing testing and quality assurance principles in to the development lifecycle.

Testing and Quality Assurance can improve quality

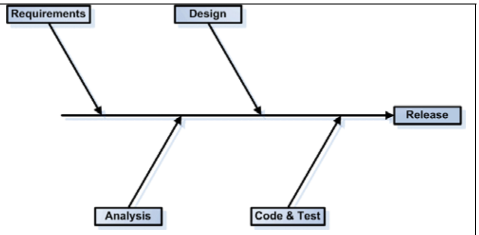
Testing should not be considered as a separate process to that of development, It should be an integral part of development, if not then testing will start later than it should and defects will be discovered much later in a product's development.
"Quality control is the process and methods used to monitor work and observe whether requirements are met. It focuses on structured walkthroughs and inspections to remove defects introduced during the software development lifecycle" (William E. Lewis, 2004).

The test and QA process that I propose incorporates the testing and quality assurance of each development stage with methodologies to prevent defect injection in to the product and to remove defects that do get injected. The defects are recorded as metrics and monitored to aid with further quality improvements.

The resulting test and QA process is intended to be adopted in any development lifecycle model. The process is wrapped around the five most typical phases of any project lifecycle and is depicted below in Figure1. A quality champion is the best person to drive such a process with a quality plan delegating certain tasks to appropriate managers or leaders.

What is behind this software process?

The test and QA process that I propose has elements of Carnegie Melon Universities Team Software Process and Rationale's Unified Process where the key project team participants are identified and for each stage there are a number of activities that these key team members have responsibility for. The key members should contribute to the process and produce a number of deliverable artefacts at each process output. Since no software product can be produced with 100% perfection. Figure 1 below depicts the two phases and five stages of the process.

The process consists of two phases: 1. Planning and design a. Requirements b. Analysis c. Design 2. Implementation phase. a. Coding & Testing b. Release	 Requirements Design Release Analysis Code & Test

In the process, the testing must befit the development effort and be planned and managed appropriately. The appriopriate test and QA steps must be selected and integrated in to the development methodology. For each development stage there must be a corresponding QA & testing stage. Prevented defect injection by quality assurance methods are also recorded. Defects must be recorded during the development stages as they are discovered. Metrics of the defects are obtained and graphed so that the quality of the software is available during the stages of development.

In addition to implementing a software improvement process, emphasis should be placed on how it is implemented. Specifically the team factors need consideration. The strength of the process depends on the experience of the team members. Their experience with the projects and with each other must be factored in.

Project factors	Team factors
• Magnitude of the project • Technical complexity and difficulty • Extent of reuse of software components • Severity of failure outcomes if the project fails	• Professional qualifications of the team members • Team acquaintance with the project and its experience of the subject domain • Availability of staff members who can support the team • Familiarity within the team members, the ratio of new people versus existing team members

Figure 1. Team Factors for process improvement.

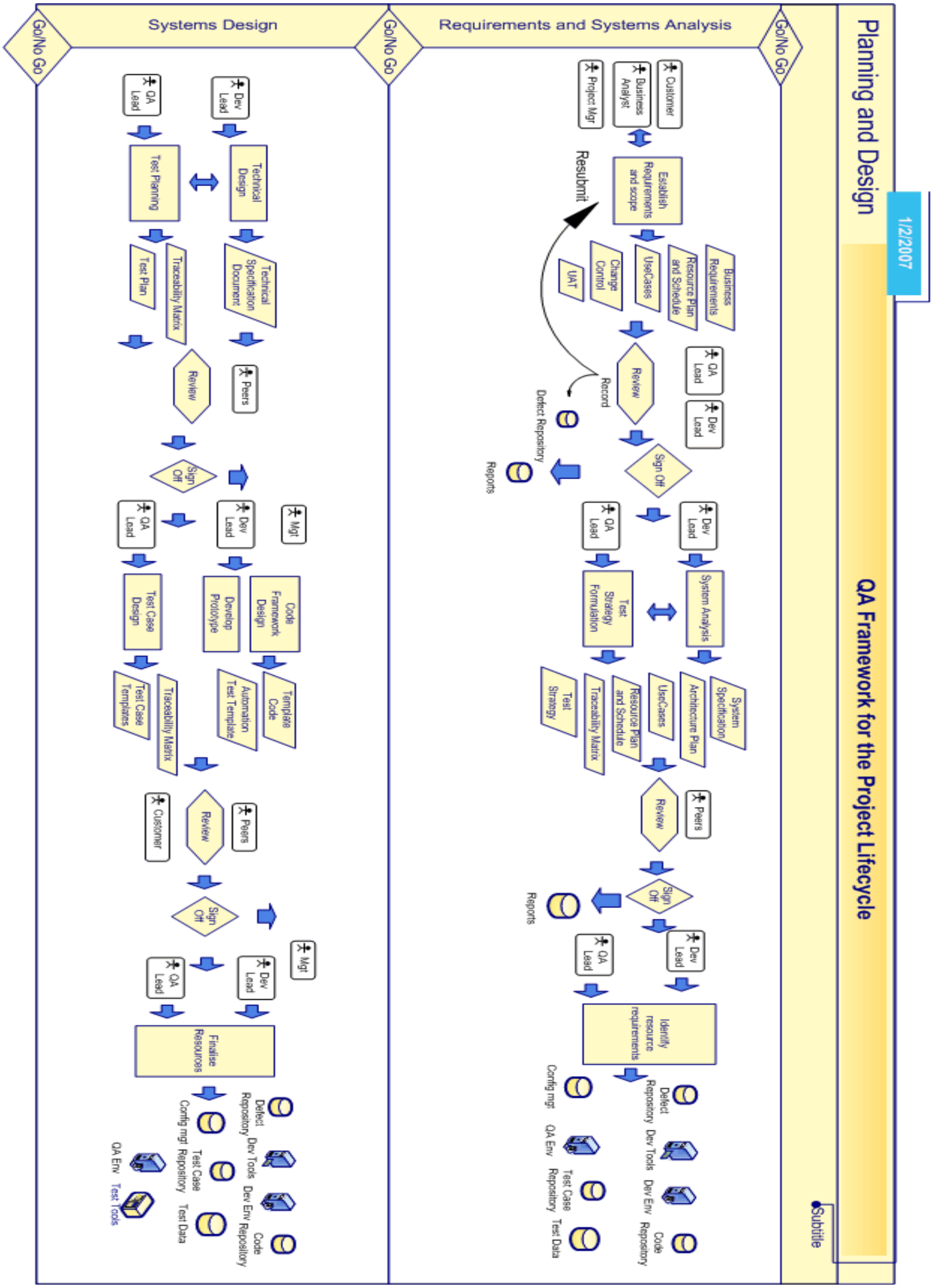


Figure 2. the QA Framework Planning and Design Phase

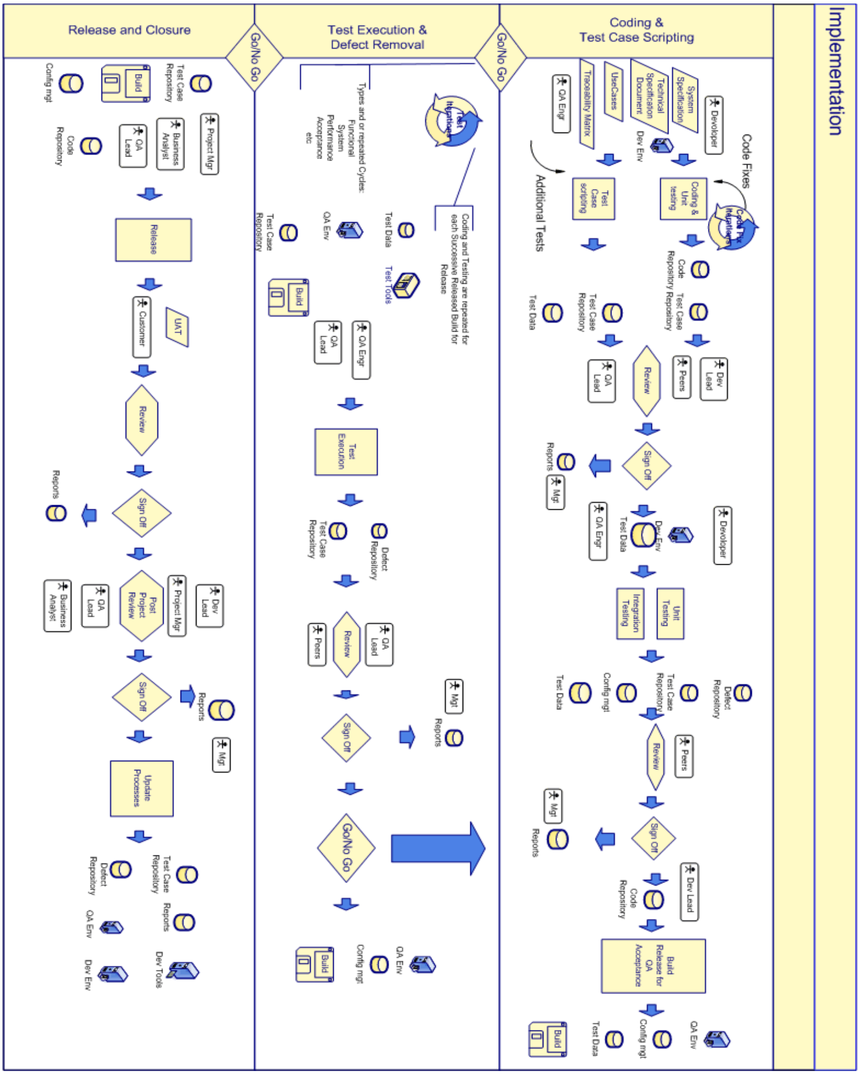


Figure 3. The QA framework Implementation Phase

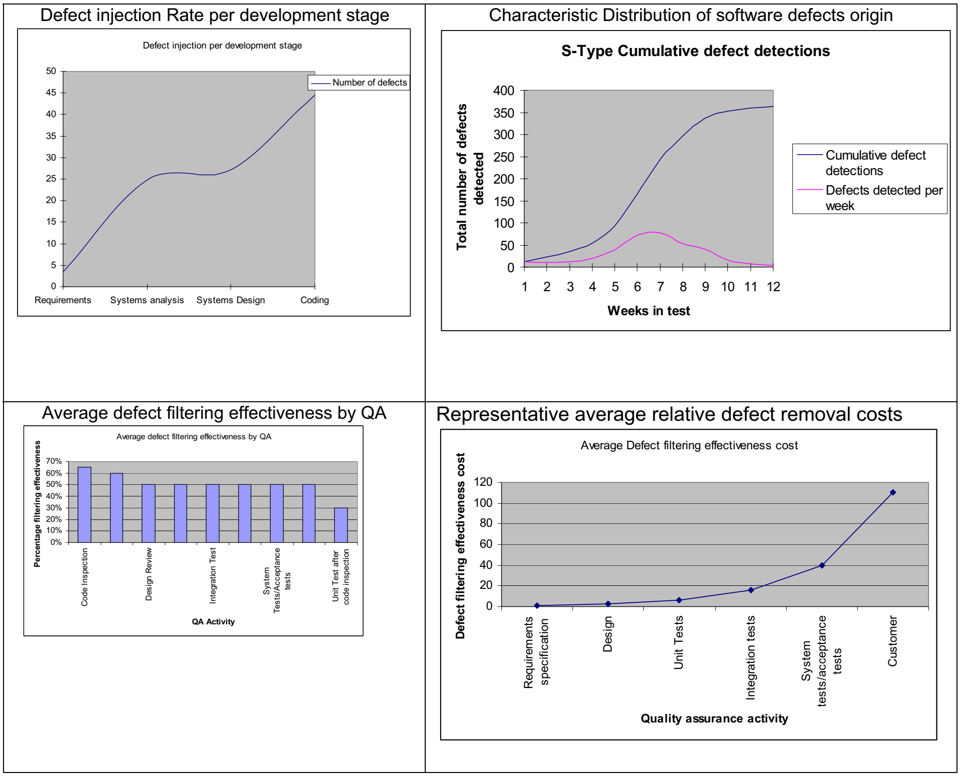
The benefits of Measuring Software defect metrics

With respect to the software, there are three classifications of software defects:

1. A Software Error is made during the development of the software
2. A Software Fault occurs as a result of an error that remains in the executing program
3. A Software Failure is a fault that results in a detectable problem

The defects can yield a lot of information with regards to the project progress and potential success or failure. Removing defects at each stage of the development process reduces the project cost while simultaneously improving quality. Preventing defects being injected into the project is more effective than removing them later. In relation to defects, Figure 6 depicts where:

1. Defects are typically injected in to projects
2. How effective QA is at detecting them
3. How expensive it is to remove them relative to who detects them
4. When good enough testing has peaked and the effort to outcome of testing is reached



Defect injection Rate per development stage

Characteristic Distribution of software defects origin

Average defect filtering effectiveness by QA

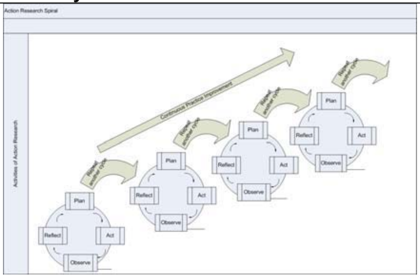
Representative average relative defect removal costs

Figure 4. Analysis of Software defects to improve quality

Process creation and implementation

The test and QA process was developed using Technical Action Research in the company where I was employed. There were two main quality issues in the company. The immediate concern was in respect to the quality of a released product and secondly with a forthcoming product. The company developed firmware and software for embedded systems. The thrust of the process improvement was in the research and development department of the company.

A proposal for improvements was placed to the company directors in relation to testing and quality assurance practice improvements. An assessment was conducted to determine the nature of the quality problems and what improvements were required. The improvements were made over three products on a three year time frame. The improvements were made using the action research paradigm.

<table>
<tr><th colspan="2">The action research cycle</th></tr>
<tr><td>1. Review the current practice
2. Identify an aspect that needs to be improved
3. Plan an improved practice
4. Act / Execution of the practice over the course of the project
5. Observe the effects of the practice
6. Reflect on the success or failure of the practice and re plan accordingly
7. Repeat the practice improvements until complete</td><td>
</td></tr>
</table>

The assessment

The assessment was planned and conducted with a summary of activities listed below:

1. Review process and project documentation.
2. Interview the senior participants of each R&D team.
3. Document the findings of each team assessment.
4. Compile and present the findings of the assessment.
5. Act on the findings and plan quality process improvements.

The findings of the assessment were documented in five categories

1. The overall defect statistics for the released product from all departments.
2. A quality report from customer support based on customer feedback on the released product.
3. Current Test case design and test planning for the product
4. An internal audit of both software, test, support and firmware departments in terms of the product project and processes
5. An assessment of the development life-cycle and quality system process in general following from the previous 4 assessment areas.

Product Quality Improvements

The first three projects (products) were conducted for the company with different development teams on projects of similar size and complexity. There were approximately 500 function points per firmware product and 1500 function points per software application and the number of lines of code was 22K for firmware and between 50K and 70K for the different software application versions.

The table below gives an indication as to the different size and complexities of the three different projects for company X.

	Project 1	Project 2	Project 3
FP	~1200	~600	~1600
KLOC	60K	22K	225K

Figure 5. Project size and complexities

Product 1 - Process improvements

The main changes to the test and QA practices for the first product included

1. Detailed test planning to identify the application components and then determining the test types to be executed for each component.
2. A risk based approach was taken to the priority of the functionality for customers and the complexity of the code that was being added.
3. A thread testing approach was taken where testing would be scheduled for the earlier modules that completed development. This was synchronised with firmware and software so that both could be tested close to the same time.
4. These tests were scheduled with milestone releases from development.
5. The tests were designed for more effective test coverage of the functionality.
6. The tests were also supplemented with detailed test data and an environment that simulated a customer's site.
7. A purposeful defect tracking tool was installed for the recording of defects.
8. The team members were assigned to the project in roles and assigned responsibilities on a par with their experience.
9. Team meetings were planned at milestone intervals to discuss project progress and for the discussion of problems from respective department perspectives.
10. Documents were devised which formalized the interaction between departments and acted as records for project progress.
11. The documentation supporting these new practices was written, circulated and approved before being placed in the companies Quality System.

Product 1 Improvement analysis

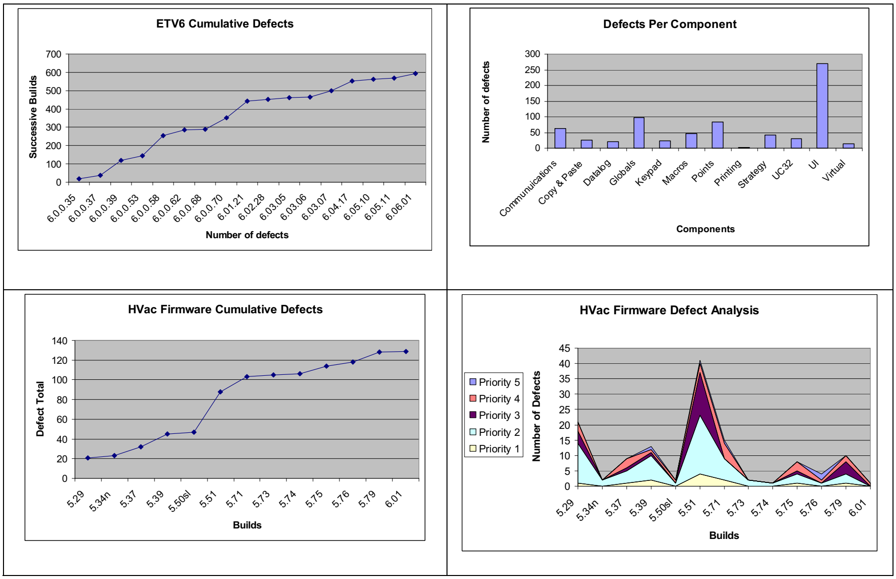


Figure 6. Cumulative Defects and defects per component

The number of defects detected during the product testing was a cause for concern. It was good that the test effort detected a high number of defects but it was an indication that the software was of a poor standard. The tail end of the curve in the accumulative graphs above indicates an upward trend towards an increase in the number of defects despite successive builds. This trend indicates that there are still defects present.

Analysis of the software application defects per component (above right) highlighted weak areas of the software application for further development work. The number of defects detected in certain components reinforced the problematic areas that the customers had experienced with an earlier release.

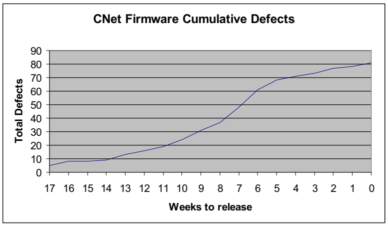
The defects for the UI were evaluated and the common causes were used for input in to future UI test case design. These tests could be executed on mock up user interfaces to save development costs in future. The most beneficial use of such UI testing would be on prototypes so that any issues would be corrected before costly backend development was undertaken.

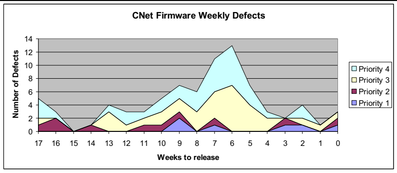
The analysis of the firmware testing indicated that there was a sporadic number of defects detected in different builds and that testing itself were not ideally executed. The analysis of the test and defect results was fed back in to the next product development process, with some additional improvements.

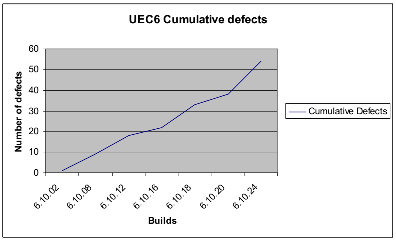
Product 2 Process Improvements

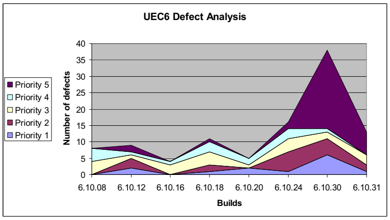
1. All previous process improvements
2. Revise and improve the UI component tests.
3. Adopt reviews and testing of UI prototypes.
4. Request the review and sign off of all development and test artefacts by customer experienced teams.
5. Implement Usability testing by end users on prototypes and beta releases.
6. Conduct full testing over three iterative cycles of tests.
7. Reuse of all previous tests, environment and data.

Product 2 Improvement Analysis









A prototype for the new UI was developed and reviewed by customers. The outcome of the review allowed functionality that was not a high priority for customer's to be omitted. This saved development time and cost. Usability of features was improved based on customers feedback.

The planned three test cycles were completed in cycles of 15, 15 and 10 days respectively. The test case design and existing data and environment had proved beneficial in reducing the test effort. High priority defects were scheduled for fixing in subsequent builds; more test passes could be achieved as a result.

There are three peaks of defects for both the software and firmware testing above. Full testing can be achieved in the three cycles. This three cycles approach is cost effective on development and test teams.
If a comparison is made between the two defect cumulative curves, there is continued increase in the number of defects detected per for the software (UEC6). There is a levelling off in the firmware (CNet Firmware) curve indicating test burnout. This levelling indicates that if testing were to continue the number of defects detected would not increase noticeably. With the software however there is no levelling off at the tail end of the curve. This is indicative of a number of outstanding latent defects in the application. The trend of the graph indicates a continued increase of defects detected in successive builds. This is the worst case scenario for testing where there was a high degree of defect injection in the design and code stages of development and that it took minimal test effort to discover a high number of defects.

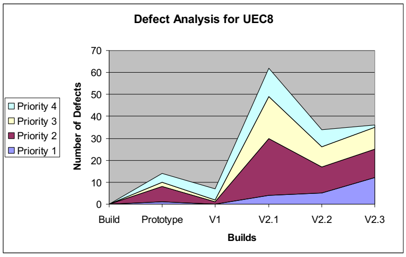
The test and QA process improved the quality of the product firmware over the course of 2 product releases while reducing the overall development and testing time. The inclusion of customers at reviews added weight to the opinion that the software quality had fundamental design problems that no amount of coding or testing could eradicate. You can not develop quality in to a product; it must be endemic in its design from start to finish.

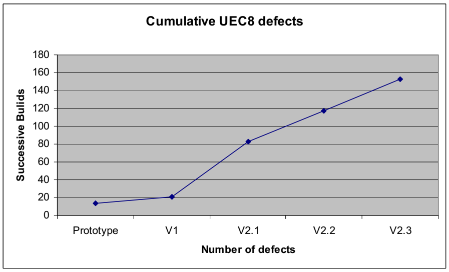
Product 3 Process Improvements

The software product was going to be re-developed offshore in India. The existing application design was used as a template for the requirements of the new version. The application was expected to be of a similar size to the existing engineering application with approximately 70KLOC. The project team consisted of 12 developers and 4 testers (offshore) over an 11 month period.

1. The testing practices, test cases and test data of the previous projects were to be re-used but in the form of User Acceptance testing since the offshore development house had to conduct their own Unit and integration testing.
2. The plan was to have three deliveries to the company with certain features delivered in succession. The three deliverables were representative of the three cycles of testing that were successful in the past.
3. The high risk areas were to be delivered first. Since this was an application heavily dependent on the UI, a prototype was to be delivered to Ireland in addition to the three staged deliveries for assessment and testing.
4. The testing schedule was risk based with high risk areas being tested first.
5. The static testing of the design documents was also planned in order to shift the focus of quality assessments to earlier in the development lifecycle.
6. The documentation was to be delivered before coding starts. The system requirements specifications (SRS) document was written and delivered followed by both high level and low level design documents (HLD – LLD) before a UI prototype and three phased deliveries of the application itself.

Product 3 Improvement Analysis





The static testing of the design documentation revealed that the design documents did not provide a logical design solution to each of the main components of the application.

The initial UI prototype (phase I delivery) which was delivered was 30 days behind the milestone delivery date. The limited testing that was possible on this prototype revealed that the UI did not offer the functionality that was required for the application. It was not until build (V2.1) that a sufficient level of functionality was present that independent testing could be conducted. This build was rejected since it had eighty three defects detected. The prototype functionality and the functionality of the first release were not present in the build until the second release V2.2. With this build the first iteration of functional testing was able to commence. This build and two subsequent builds were rejected on the grounds that the severity and number of defects was below the permitted quality level. The number of serious defects was increasing with each successive build (4, 5 and 12 respectively). In The number of defects increased with each successive build from the contractors. The quality of the software produced was below that of the old poor quality software. The test effort was in the low effort and high output bracket where a large number of defects were detected with minimal effort. This is the worst case scenario for software quality. The test effort in my company was UAT. The Unit, integration and systems testing performed by the contractor was below an acceptable level.

A code review was conducted on the delivered code to provide secondary evidence on the standard of software. The code review corroborated with the findings of the testing, that the standard of code was sub standard. Based on the test results and code review a decision was made to terminate the contract and cancel the development project. This early termination saved both time and money for the company.

Summary

Software quality is not easy to achieve. It is only possible with building quality into a product from the outset with dedication to quality from all team participants. A quality champion is required to drive the quality process throughout the product development. Software quality is a continuous process where successive improvements are made at each project stage and subsequent project.

The implementation of a quality improvement process requires consideration of the team participants and input from all team members. The customer is the most important team participant in the quality program. The customer approves the requirements at the beginning of a project and takes delivery of the product at the end.

Throughout each stage of development, testing and quality assurance methodologies can be implemented to determine the quality of each stage deliverable. Metrics should be obtained to determine the quality of the product at each of these stages. By graphing these metrics, analysis of metrics will indicate where quality problems lie, and where improvements can be made.

The testing and quality assurance process is ideally an integral of the development process. With a strategy for quality improvement in place and with a suitably experienced helm, working with a dedicated team, only then can sailing be fulfilling no matter what the weather!

TIPA to keep ITIL going and *going*

Marc St-Jean
Centre de Recherche Public Henri Tudor
29, avenue John F. Kennedy,
L-1855 Luxembourg
marc.st-jean@tudor.lu

Abstract

This industrial case presents how an ISO/IEC15504 IT Service Management (ITSM) assessment can be used as a managerial tool for setting measurable goals that will improve in a sustainable way the IT service management processes and serve as a motivational tool for IT Service employees with a case illustration of how it was used at the CRP Henri Tudor.

Keywords

ISO/IEC15504, ITIL, ITSM, process assessment, management, motivation

1 Introduction

Since Henri Fayol defined the basis of modern management in 1916 [1], control has been a key element of the manager's role. Control as defined by Fayol is the fifth function of management after planning, organizing, commanding, and coordinating. It is an important function in any managerial process as planning requires fixing measurable objectives. Without any control it is impossible to say if previous decisions have been beneficial or not to the enterprise. Tudor's ITSM Process Assessment, previously known as AIDA [2] [3] [4] [5] [6] [7], is offering an objective methodology based on the ISO/IEC15504 [8] to control the IT Service Management

2 What is TIPA?

Tudor's ITSM Process Assessment (TIPA) is a methodology that provides a framework and concrete steps to perform a capability process assessment of IT Service Management processes. TIPA follows the ISO/IEC15504-2 standard approach (formerly known as SPICE[1]) to perform a compliant process assessment of IT Service Management processes in order to determine process capability. "This framework can be used by organizations involved in planning, managing, monitoring, controlling and improving the acquisition, supply, development, operation, evolution and support of products and services." [9]

The CRP Henri Tudor applied the process assessment ISO standard to the best practices of IT Service Management as defined by the IT Infrastructure Library® (ITIL®) [10] [11]. The latter was developed by the Office Government Commerce, a British independent office of Her Majesty's Treasury, which was created to „help Government deliver best value from its spending" [12]. It offers an effective way of engineering the IT services. As the efficiency gains from implementing an ITIL framework are

[1] Software Process Improvement and Capability dEtermination

very high – an IDC study [13] concluded that the application of ITIL brings in average an overall efficiency gain of 30% and a ROI over a three years project life cycle of 422% – more and more companies start to manage actively their IT department. ITIL is today a worldwide recognised set of best practices in the field of IT Service Management.

TIPA thus offers a standardized methodology to assess IT service management capability which is objective and thus repeatable. It gives a precise assessment as a certain point in time which can be used for benchmarking or for goal setting in a continual improvement process. TIPA also includes a SWOR (Strength, Weakness, Opportunity and Risk) analysis which provides the basis of an improvement plan and recommendations. It is easily scalable to the goals, the size and the industry of the organisation.

The TIPA methodology involves interviewing a mix of service providers (technicians and managers) and final users. This provides the data required to understand how the different ITSM processes are managed and performed. This technique also offers the advantages of reminding the employees the importance of the service component of the internal IT in an enterprise.

3 Presentation of the CRP Henri Tudor

The Public Research Centre Henri Tudor (hereafter referred to as CRPHT) is the largest Research and Technology Organization in Luxembourg. Thirty percent of its total income is a grant for mission of public utility, and the residual funding relies on European or national R&D programmes or comes from projects directly funded by the private sector. CRPHT has over 300 employees, among which 83% have a Master’s degree or a PhD. Staff expenses represent almost 60 % of the total expenditure of CRPHT. The CRPHT is run as a private limited company. Since its inception in 1987, CRPHT has experienced a significant growth and is sometimes compared to a start up of “public interest”.

CRPHT pursues two main strategic objectives. First, it carries out R&D activities in a collaborative way in order to improve and strengthen the innovation capabilities of private and public organizations. Second, it aims at being a recognized scientific player in selected technological and scientific areas.

The Centre for IT Innovation (hereafter referred to as CITI) is a department of the CRP Henri Tudor, which mission is directly derived from the two strategic objectives of the CRPHT. In order to achieve its missions, CITI is involved in the various stages of the innovation chain: from doctoral research to technological and methodological assistance, coordination of innovation networks or professional and scientific conferences.

The CITI is engaged in numerous ISO committees and is involved since many years in areas in and around the ISO/IEC15504 and ISO/IEC20000 [14] [15]. More specifically the CITI is one of the main players in Luxembourg Standardization Committees such as Sub-Committee 7 (SC7 - Software and Systems Engineering) and SC27 (Information Security), within the ISO Joint Technical Committee 1 (JTC1) dealing with Information Technology. Two CRPTH’s employees are also co-editors respectively of two parts of ISO/IEC20000: the future part 4 ITPM Process reference model [16] [17] and the future part 5 Exemplar implementation plan for ISO/IEC20000-1. Building on this experience, TIPA was developed at the CITI, whom was the first to use the ISO/IEC15504 framework to develop an ITIL process assessment.

As this paper will use the CRPTH’s case to demonstrate the added-value of TIPA’s framework, it is important to consider that the CRPHT has grown relatively fast in the last 10 years, more than doubling its staff, and started implementing ITIL for its IT service management in 2004. The IT Service department consist of 7 full-time employees and covers the three locations of the CRPTH in Luxembourg.

4 The assessments

Two assessments were performed at the CRP Henri Tudor. The first one was made in 2005 and the second one at the end of 2008. A first assessment of ITIL processes was made in 2005 and a second one in December 2008. As the goal of this case is not to highlight the value of ITIL to improve the performance of IT and its impact on the company, but to determine the added value of the process assessment, the result of both assessments will not be studied in details.

The first step of the assessment is to determine its scope. It was decided with the IT Service Director, to assess Incident Management, Change Management and Configuration Management processes. The assessment consisted of 13 interviews of the IT service employees and users (of the services). As per TIPA's methodology, two assessors performed the interviews and rated each process. As it was one of the first assessments ever performed by the CRPHT, it took more interviews and more time consolidation the data than in the following assessments. The final assessment contained short and long term recommendations that were to be implemented in the following years. Based on this report, the IT Service also set new goals for the coming years. The second assessment performed in December 2008 covered the same three processes plus Service Level Management and interviewed 7 IT service employees and users. We have to note that the second assessment was performed much faster as both the assessors and assessed knew where they were going i.e. less time was required explaining how the assessment would be performed and the scope of the assessment was easier to set.

5 A managerial look at TIPA

As mentioned in the introduction, control is an important element in management. In any planning, one has to set goals to achieve and these goals normally have to be measurable. Management by objectives [18] is still one of the basis of modern management and it does ask for measurability. Thus TIPA, or for that matter any ISO/IEC15504 process assessment, serves the purpose of measuring the advancement of what was previously planned. This is important from a management perspective to measure if the investment in the project has paid off, if the goals were achieved, surpassed or if there is still a lot of work to do to get there. Moreover, managers are normally cautious to engage in a project where they will not be able to measure its advancement or to set the goals. In this case, implementing ITIL and a continuous improvement process cost money and, without tools like TIPA, it is difficult to measure. First because ITIL is fully scalable, one does not have to implement the whole best practices. For example at the CRPHT, which is a relatively small enterprise, we decided to adopt certain practices but not all of them. Measuring the capability of the processes versus measuring if the best practices were put in place gives more freedom to the IT Service Managers. They can then define how they will achieve their goals instead of having imposed on them a way to do, which might not be aligned on their goals in relation with their industry, size and specific conditions.

For example, one process defined by ITIL is Incident Management, which has for goal to restore normal service as quickly as possible and to minimise impact on business operation. A typical measurable goal could be to reduce by 20% the down time per year, but this goal could be achieved in many ways. One way would be to add employees to solve the problem when it happens. Another way would be to study past incidents to diminish the risk of recurrence. Two very different approaches, one which will increase the cost of IT Service Management and another one which might in the long run reduce them. ITIL is offering the best practices to improve the efficiency and efficacy of the IT Service Management by offering practices, to put in place a sustainable improvement of IT Service Management. The problem with ITIL when managers plan and decide where to invest (or divest) money is that it is much easier to set a goal like cutting the down time by 20% and the budget by 10% than having a fuzzy and immeasurable goal of implementing ITIL's Incident Management Process (Incident detection and recording, Classification and initial support, Investigation and diagnosis, Resolution and recovery, Incident closure, Incident ownership, monitoring, tracking and communication). The goal is fuzzy because these processes are probably already there in a way or another. For example the classification and initial support is surely done but the classification might be done mentally by the technician answering the phone and it might not be defined nor monitored – thus making it harder to identify problems.

TIPA offers a very good compromise in the sense that a manger can now set a measurable goal of achieving in two years the Established (3rd) level of capability in the process, meaning that the process implementation of Incident Management (to follow on the above example) should be planned, monitored and well defined. This offers goals that will improve not only the end result like reduction of downtime and cost reduction but should also lead to higher capability in the process until the continuous improvement level is achieved.

As TIPA offers an objective methodology to assess the capability level, it also serves as a motivator for IT Service employees whom know there effort will be evaluated in global maturity and improvement mind-set. Goal-Setting theory [19] has put goals at the centre of organisational psychology; it has been showed that well defined goals make employees work harder to achieve it. As highlighted in an interview with the IT Service Director of the CRPHT, without this goal setting aligned on capability of ITIL processes, IT technicians tends to focus on their speciality, which might be database administration or network management for example but never IT service. Setting a goal aligned on ITIL and knowing it will be assessed forces them to think outside the box and to start thinking as service managers. It is also important in this equation that TIPA is perceived as neutral and objective, as from a motivational theory perspective this is an important variable: if employees feel that the goal evaluation is not objective and that the boss has the final word, goals can become a source of demotivation for employees.

6 The impact of TIPA at the CRP Henri Tudor

An interview with the IT Service Director confirmed that TIPA has been a catalyst of ITIL implementation at the CRP Henri Tudor. He says that without TIPA is was very difficult to set ITIL goals. TIPA did not only serve to set the goals in measurable terms but also helped to define them in relation with the size and mission of the centre. The fact that TIPA involves interviews of both sides of the service, providers and users, gave a more service oriented assessment than they could have performed. This has been very useful in creating a service culture in the IT department.

Setting clear and measurable goals in the implementation of ITIL has also helped to gain top management support. It is easier to sell the investment when top managers can be reassured that advancement will be measured objectively so that they can follow it without having an ITIL expertise.

Another point the IT Service Director highlighted is that the assessment was made by people from another unit that were both assessors and users and the feedback from different users (interviewed in the assessment process) had a great value to attach their work to the end-user - too often forgotten by the IT technicians. He reinterred the fact that having this feedback from users, especially presented in an ITIL framework, was a great factor of motivation for the IT service employees, as it does point out what was well made and what improved, but also pinpoints what has to be improved.

On the motivational front, the IT Service Director also says that TIPA is of a great help to keep people working on ITIL and consequently working on the improvement of the IT Service management. The IT Service Director says: “We are a relatively small organisation, in larger one there are people dedicated full-time to ITIL, here it is only one thing on our mind as we cannot afford someone working full-time on it. Thus TIPA is a good tool to force ourselves to do something about it. In fact, he adds, without TIPA our implementation of ITIL would probably go nowhere as there is always something more urgent in an IT Service department than working on the implementation of best practices...”

The IT Service Director also indicates that TIPA identified problems touching other departments, one example being the exit management of employees where the roles and procedures shared between the workers unit, the HR department and the IT Services were not well coordinated.

Finally, the IS Service Director adds that TIPA is a useful assessment for his service. He is the one that asked for a second one and he says he would love to replace the annual ISO9001 audit by a TIPA assessment. The reasons: as our auditors of the ISO9001 are not specialists of IT Service Management, the audits are more difficult as the auditor and the IT department speak, in a sense, different languages. While with Tudor’s ITSM Process Assessment, the questions are much more pertinent and the conversation is much easier as they do “speak the same language” and have similar goals. He adds that the ISO9001 audit is mostly useless to him when compared to TIPA. Not only does the audit just point out to details that have not respected the procedure he wrote, but it does not have any added value when you make a TIPA. The latter point to the ITIL de facto standard, which, says the IT Service Director, is an amazing tool to improve quality.

7 Conclusion

It cannot be denied that managers and employees like to have clear goals, which can be measured objectively, thus in that frame of mind a process assessment like TIPA, is a great tool to help setting goals and to assess the advancement towards them. It can also be a great tool to keep an ITIL project and the associated motivation going. As ITIL has been shown to improve considerably the efficiency and efficacy of the IT services – offering a very good return on investment – the process assessment of ITIL, TIPA, can be an excellent catalyst to ensure that the project stays on track.

Literature

1. Fayol, H. (1916), "Administration industrielle et générale", Bulletin de la Societe de l'Industrie Minerale, Vol. 10 No.3, pp.5-162.
2. Barafort B., Di Renzo B., Merlan O.: Benefits resulting from the combined use of ISO/IEC 15504 with the Information Technology Infrastructure Library (ITIL). Proceedings of the International Conference PROFES'2002, Rovaniemi, Finland, 2002.
3. Barafort B., Di Renzo B.: Assessment and improvement integrated approach: combined use of the ISO/IEC 15504 (SPICE) and the Information Technology Infrastructure Library (ITIL), Proceedings of the National Conference SPIRAL'2004, Luxembourg, 2004.
4. Barafort B., Di Renzo B., Lejeune V., Simon J-M. : ITIL Based Service Management measurement and ISO/IEC 15504 process assessment: a win – win opportunity. Proceedings of the 5th International SPICE Conference on Process Assessment and Improvement, Klagenfurt, Austria, 2005.
5. Hilbert R., Renault A.: Assessing IT Service Management Processes with AIDA – Experience Feedback. Proceedings of the 14th European Conference for Software Process Improvement EuroSPI, Potsdam, Germany, 2007.
6. Béatrix Barafort, David Jezek, Timo Mäkinen, Svatopluk Stolfa, Timo Varkoi, Ivo Vondrak: Modeling and Assessment in IT Service Process Improvement . Proceedings of the 15th European Conference for Software Process Improvement EuroSPI, Dublin, Ireland, 2008.
7. www.aida-itsm.lu
8. ISO, ISO/IEC 15504-2: Information technology - Process assessment - Part 2: Performing an assessment, 2003.
9. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=37454
10. IT Infrastructure Library – Service Delivery, The Stationery Office Edition, ISBN 011 3308930, 2001.
11. IT Infrastructure Library – Service Support, The Stationery Office Edition, ISBN 011 3308671, 2000.
12. http://www.ogc.gov.uk/about_OGC.asp
13. IDC, "Determining the return on Investment from Deploying Integrated IT Service Management", April 2006 – found in Service Management Implementation Approach published by Pink Elephant, 2008
14. ISO, ISO/IEC 20000-1: Information technology – Service management – Part 1: Specification, 2005.
15. ISO, ISO/IEC 20000-2: Information technology – Service management – Part 2: Code of practice, 2005.
16. ISO, ISO/IEC JTC1/SC7 3797, NWI Proposal - Information Technology - Service Management Process Reference Model, 2007.
17. ISO, ISO/IEC JTC1/SC7 3798, NWI Proposal - Information Technology - Process assessment - Part 8: An exemplar process assessment model for IT service management, 2007.
18. Drucker, Peter F., "The Practice of Management", 1954
19. Locke, Edwin A. (1968) "Toward a Theory of Task Motivation and Incentives" Organizational behavior and human performance, (3)2: 157-189.

Interview: Hugues Henriot, IT Service Director of the CRP Henri Tudor, Luxembourg, 11th March 2009

Author CVs

Marc St-Jean

Marc St-Jean has extensive experience in management consulting and is titular of a Master in Business Administration (MBA) from McGill University, Montreal (Canada). He has taught in Canada Organisational Behaviour and general business in college. He now works in Luxembourg for the CRP Henri Tudor as project manager of TIPA.

Process improvement in IT departments: a four years experience in a large retail company

Paolo Salvaneschi[1], Daniele Grasso[2], Maurizio Besurga[2]

[1] University of Bergamo, Faculty of Engineering, Dalmine (BG), Italy
and Salvaneschi & Partners, Bergamo,Italy
pasalvan@alice.it

[2] Mediamarket S.p.A, Curno (BG), Italy
dgrasso@mediaw.it
mbesurga@mediaw.it

Abstract

This paper describes a four years experience of process assessment and improvement for the IT department of a large retail company. The involved company and the organization of the IT department are shortly presented. A first process assessment was done during 2005. We describe the methodological approach, based on ISO 15504 and the assessment results. We also discuss both the improvement actions and the results of a second assessment, done in 2008, presenting successes and difficulties. Finally, we highlight benefits, problems and lessons learned.

Keywords

Software process assessment, Software process improvement, IT Departments

1 Introduction

Software process assessment and improvement are significantly used in ICT industry and have been a significant research issue [Fuggetta 2000]. Case studies have been published for large ICT organizations [Lindholm 1998] [Lamprecht 1998] [Mc Caffery 2008]. The literature also provides case studies [Bucci 2000] [Calvo-Manzano 2002] and specific approaches [SPIRE 1998] for small and medium enterprises.

Software process assessment and improvement is also important for IT departments of non-ICT organizations, because their business, in many cases, depends significantly on software-related processes managed by the CIO. The quality of such processes (in house development, acquisition processes as well as service delivering processes for internal users) is important for the performance of the whole organization. This is the reason why the methods for improving processes may be usefully applied to IT departments of the client side in the ICT market.

Nevertheless, to the best of our knowledge, a limited set of case studies has been published. Armbrust [Armbrust 2008] shows an experience of tailoring and deploying German V-Modell XT process standard in the IT department (80 employees, doing the majority of the development work internally) of a medium-sized (2,200 employees) company in the mail order business, part of a larger group. Other experiences have been reported for specific processes like software testing in a large commercial banking organisation [Downey 2008] or requirements engineering process in different types of organisations [Sommerville 2005].

In 2006, we published an experience report [Salvaneschi 2006] presenting the process assessment done during 2005 and the first improvement experiences for the IT department of a large consumer electronics retailer. In this paper, we want to complete the case study describing the whole improvement process along four years, the results of a second assessment done at the end of 2008 and the lessons learned.

Section 2 shortly introduces the company involved in the case study and the organization of the IT department. Section 3 presents the methodological approach of the assessment, based on the ISO 15504 [ISO15504] norm and the assessment results. Section 4 discusses the improvement actions. Section 5 presents the result of the second assessment, while the last section discusses benefits, problems, lessons learned and draws some conclusions.

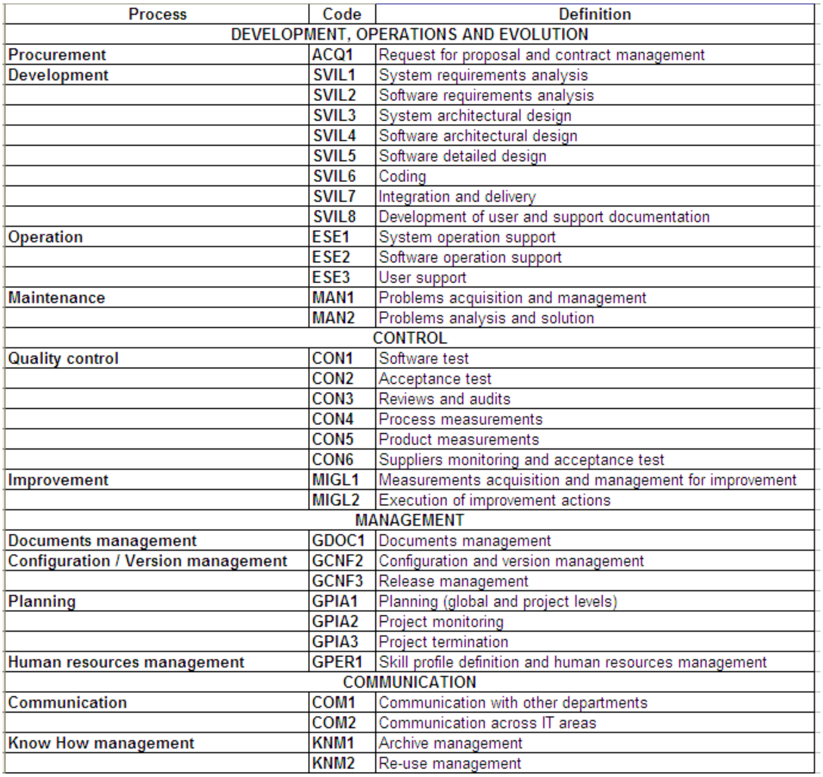
2 Case study

The Mediamarket company is part of the Media-Saturn Holding GmbH, the European leading company in large-scale retail of Consumer Electronics and Domestic Appliances with a total of 702 stores in 15 European countries, more than 17 billion Euro turnover and more than 48.000 employees (year end 2007). The Italy based company manages the brands MediaWorld and Saturn and operates 88 stores in the whole country, with 2 billion Euro turnover and more than 7000 employees (year end 2007). The IT department manages a net of about 150 servers and 3000 clients and hosts about 100 software applications delivering services to the users. The IT Department (about 70 employees) is structured in six areas: AS400 applications, Intranet Web-based applications, Data Warehouse / Business Intelligence, Data Centre, Telecommunication and Help Desk. The first three areas are development-oriented. They partially write software internally, while suppliers develop the majority of applications or tailor market-available packages for the specific needs. The last three areas are service-oriented and provide services to the other business units of the company.

In 2005, the IT Management decided to run an improvement activity of the IT processes, starting from an initial assessment. The aim of the assessment was to provide support for a better understanding of the current state of processes and a more conscious allocation of effort for improvement. The driver for the management decision was the fast growing process experienced by the IT department and the need to reach a new stability level for supporting a higher service level for internal clients.

3 First assessment

The assessment process is based on ISO 15504 [ISO 15504] standard and is composed by the following steps: selection of interesting processes; gathering of evidences and measurement of processes quality profile; business goals identification and definition of the required quality profile; diagnosis and suggested actions. The identification of interesting processes is based on ISO 15504 and ISO 12207 [ISO12207] sets of processes. Table 1 shows the tailored list of processes.



Process	Code	Definition
DEVELOPMENT, OPERATIONS AND EVOLUTION		
Procurement	ACQ1	Request for proposal and contract management
Development	SVIL1	System requirements analysis
	SVIL2	Software requirements analysis
	SVIL3	System architectural design
	SVIL4	Software architectural design
	SVIL5	Software detailed design
	SVIL6	Coding
	SVIL7	Integration and delivery
	SVIL8	Development of user and support documentation
Operation	ESE1	System operation support
	ESE2	Software operation support
	ESE3	User support
Maintenance	MAN1	Problems acquisition and management
	MAN2	Problems analysis and solution
CONTROL		
Quality control	CON1	Software test
	CON2	Acceptance test
	CON3	Reviews and audits
	CON4	Process measurements
	CON5	Product measurements
	CON6	Suppliers monitoring and acceptance test
Improvement	MIGL1	Measurements acquisition and management for improvement
	MIGL2	Execution of improvement actions
MANAGEMENT		
Documents management	GDOC1	Documents management
Configuration / Version management	GCNF2	Configuration and version management
	GCNF3	Release management
Planning	GPIA1	Planning (global and project levels)
	GPIA2	Project monitoring
	GPIA3	Project termination
Human resources management	GPER1	Skill profile definition and human resources management
COMMUNICATION		
Communication	COM1	Communication with other departments
	COM2	Communication across IT areas
Know How management	KNM1	Archive management
	KNM2	Re-use management

Table 1: Interesting processes for the IT Department

They include the whole set of ISO 12207 processes with some rewording and variations needed to be more coherent to the IT department characteristics and vocabulary. Table 1 shows four groups of processes. The first two groups include customer-supplier and engineering processes (according to ISO 15504 classification). They take into account both system and software aspects. Specific relevance is given to the quality control activities. The third group is related to project support and organizational processes. The last group includes activities (partially outside the ISO definitions) related to the establishment of a good communication practice both inside the IT department and between IT department and business users. It also includes the knowledge management process required for maintaining and reusing the knowledge owned by the IT department.

Table 1 lists all the considered processes, while each area is involved in a subset of them. The process capability level is based on the ISO 15504 classification:

NA Not interesting for the specific Area;
0 Not performed or partially performed;
1 Performed informally;
2 Planned and tracked with documented evidences;
3 Based on standard and tailored methods;
4 Quantitatively controlled;
5 Continuously improving.

Information is collected and processed through the following steps:

- Kick off meeting with IT manager, area managers and assessor. The project is presented and discussed. A plan of actions follows.
- The assessor meets both area managers and representative people, collecting the available information (definition of interesting processes; classification of projects / activities; selection of significant examples for each class; gathering of available documents for each example; description of current practices)
- The assessor analyses the information (using the ISO 15504 practices as guideline), discusses the results with the area managers and presents the final report to the IT manager. A final meeting involves all the assessment participants.

Figure 1 (left side diagram) shows the measured profile for one of the software development areas (see table 1 for process codes). The main part of the development processes is planned with documented evidences (Level 2). Operation and maintenance processes are at level 1. The main part of quality control processes is at level 1. Support and communication processes are at various levels. Document and configuration / version management are very well performed al level 3, while some other processes (e.g. planning) require significant improvements. The large differences in process maturity are mainly due to business priorities as well as time pressure and locally available skills. The right side diagram shows the measured profile for one of the service-oriented areas. If we look at the complete assessment, we see, in most cases, similar diagrams for the remaining areas of the two classes (development oriented and service-oriented).

Twenty people and about thirty projects were interviewed. The assessment took fifteen days.

The IT manager evaluated the assessment report for identifying the critical issues and for defining the improvement goals, according to the management drivers mentioned on chapter 2. The rationale behind the selection of the improvement profiles is essentially the need that every interesting process should be at minimum at level 2. GDOC1, GCNF2 and GCNF3 processes in the left side diagram are already at level 3, but this is not true for all areas. Moreover, two groups of processes more oriented toward supporting service and evolution (knowledge management for all the areas as well as operation oriented processes for service oriented areas) should reach level 3.

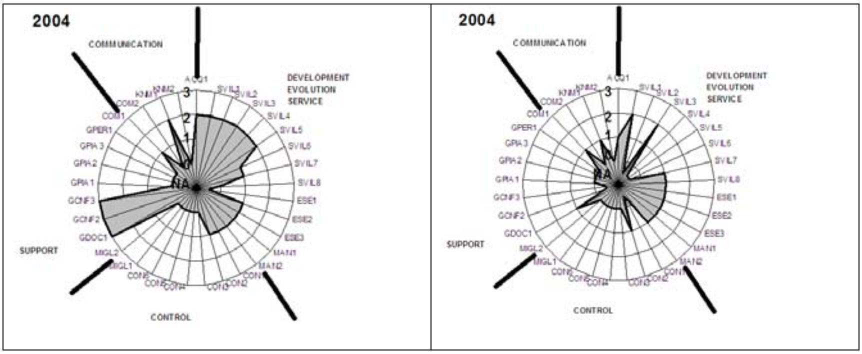


Figure 1: Measured process profiles for a software development area (left) and for a service-oriented area (right)

The result is a required quality profile for the processes of each area. For example, figure 2 shows the measured profiles referred in figure 1 (grey area) and the required improvements (black area). The required quality profiles of figure 2 are examples. If we examine the whole set of quality profiles, the required improvements are similar for each profile class (development oriented and service-oriented). Each area may have in addition some specific required improvements. More details about the assessment methods and results may be found in [Salvaneschi 2006].

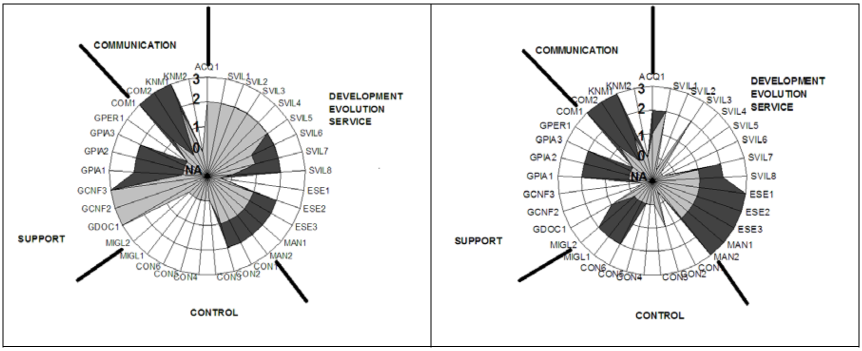


Figure 2: Required vs measured process profiles for areas referred in figure 1.

4 Process improvement

The assessment report includes detailed evidences and examples that are the basis for assigning process capability levels. These evidences support the explanation of the measured quality and the identification of improvement actions. The first assessment outcome was an improvement plan that progressively involved the selected processes. The plan was released in 2005 and has been implemented (with the needed modifications) during the following three years.

We shortly present the main actions:

- Identification of development processes and related standards of documents (We modelled two types of processes, for small and large projects, and we delivered templates for all the released documents. We also developed specific adaptations (for example for Business Intelligence projects);
- Quality control. We defined the testing and acceptance process as well as the associated methods and tools. It was involved an external supplier that now provides the functional testing of each software application before the delivery. The testing service acts as an independent party both for internal development teams and external software providers.
- Document and version management. We defined the structure of the central document database collecting, under version control, all the documents released by a project. The database stores both service-oriented and development-oriented documents, including source code. A configuration and version management tool supports the whole process.
- Help desk and maintenance support. The process was reorganized through the extensive use of a ticket management tool. The Help Desk area acts as first-line for problem solving and forwards selected tickets to the development and other service areas. These tickets may become change requests for development areas.
- Planning. It was defined a planning process implemented through a project planning tool. Each area manager was involved in the responsibility for using the tool, with a central support at IT Department level.
- Knowledge management. The central document database was divided into two parts. The first one stores all the project-related documents, while the second stores all the application-related ones.

The key concept is the difference between projects and applications. An application is a delivered system (composed by hardware, software and services) providing service to internal clients. An application is typically a long living system (for example, it may be in service and may evolve for ten years). A project is an allocation of resources for producing a new application or evolving one or more existing applications. The typical project life is one year or less. It was defined a process for managing the two types of documental structures and for updating the application documents at the end of a project. Moreover, the application part of the database was loaded with part of the most critical information concerning the legacy systems (both for development and service areas). The main effort was the development, with a tool support, of the data models for the AS400 operational database and the Data Warehouse database.

All the improvement actions reported directly to the CIO and involved an external consultancy organization. The improvement focused on a subset of processes at a time, moving to another subset only when the previous one upgraded to a new reasonably stable level. All the people were involved with a periodical monitoring of the achieved results.

5 Second assessment

The whole improvement process was periodically monitored for evaluating the progress and deciding the required actions. At the end of 2008, the IT management requested a second assessment for evaluating results, benefits and problems.

The assessment was realized using the same methods of the first one, for producing comparable results. Figure 3 shows the new measured profiles for the same software development (left side diagram) and service-oriented (right side diagram) areas represented in fig 1. The comparison between 2004 and 2008 measures for these two areas is a good sample of the whole set of areas, even if some variations exist.

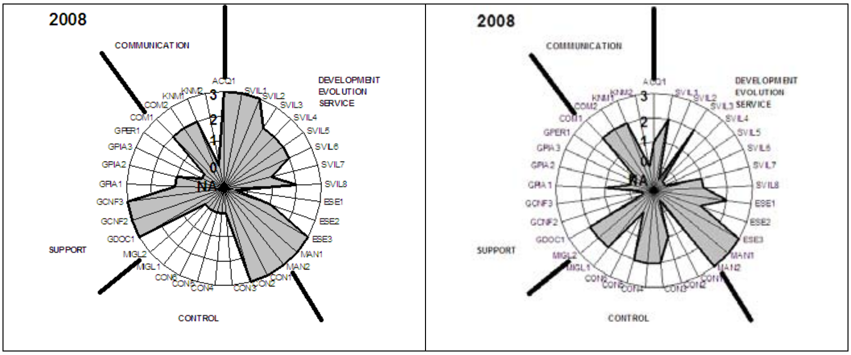


Figure 3: Measured process profiles for areas referred in figure 1 – second assessment.

Development processes (left side diagram) moved partially from level 2 to level 3. In the other development areas, this upgrade was related to both requirements and design processes, while this area experienced more problems in design processes. Note the diagram shape concerning the process SVIL7-Integration and delivery. This shape is the same for all the three development areas. The delivery process is again an open problem.

Maintenance and quality control processes improved more than the planned goal, from level 1 to 3. Configuration and document management processes remained at the same good level. The other two development areas improved these processes, moving from level 1 to 3.

As we can see from the diagram, planning process where not able to reach the stated goal. Knowledge management and communication processes improved significantly from level 1 to 3.

We will not describe at the same detail the changes in the right side diagram, related to a service area. The global shape shows significant improvements, even if not all the stated goals are reached.

6 Benefits, problems and lessons learned

An important question is about the benefits of the improvement effort. Did the improvement effort deliver significant benefits to the company business? In the following, we will provide qualitative as well as quantitative evaluations for answering the question.

A first consideration is that the process reengineering, caused by the assessment and improvement activities, supported the ability of the IT department to follow the growing business needs. The number of active projects changed from 40 in 2005 to 120 in 2008, while the managed memory space moved from 5 Terabytes in 2005 to 12 in 2008.

Help desk and maintenance support moved from about 500 managed tickets per month to about 4000, completely managed through the support tool.

The quality control process provided the functional test of about 45 applications including complex distributed applications as well as evolution projects with impact on many existing applications. This testing activity delivered about 6000 test cases. The 9 % of them revealed a serious malfunctioning. Obviously, we cannot say that the quality control process prevented to deliver about 550 malfunctioning functions to the company users. Even a less engineered process would have revealed a number of malfunctioning functions. Nevertheless, it is reasonable to say that the testing procedure with tailored testing strategies and defined test coverage criteria, improved the failures detection, particularly for the most dangerous malfunctions.

The development processes and the service delivery processes received a benefit, both for internal efficiency and ability to support the company business, from the well-ordered management of a large amount of documents. These documents describe not only IT internal information (for instance software design) but also information with direct impact on business like for example business requirements or documents for service delivery support. In this case, we are not able to provide numbers, but only a qualitative evaluation.

A second significant question is about where we had problems and why, and what are the areas needing further improvement.

We had problems in the planning process. After the first delivery of the process and the associated tools, we experienced a difficult use, due to the limitation of available human resources as well as to the difficulty both to plan large projects and to react to numerous smaller business requests. Now the process is in a revision phase.

The knowledge management process made significant advances, but needs more improvement. The type and structure of documents required for supporting the evolution of applications must be better identified for reducing the maintenance cost. Moreover, the applications delivery process must be improved.

Finally we list and comment in the following the lessons learned during the experience:

- Improvement requires time.
  It is difficult to implement an improvement process in a short time. Each improvement action requires an extra effort added on top of the day-by-day job. This is true in the transition phase until the new way of doing the job is stable at the new improved level.
- Improvement requires continuous effort, monitoring and commitment.
  It is not sufficient to define and explain a new process or tool. Managers must monitor the acceptance of the proposed change and support the real adoption.

- IT manager sponsorship as well as people involvement and motivation are mandatory.
  Without a strong commitment and direct involvement of the IT manager, the improvement project is not feasible. Moreover, the IT manager must motivate all the involved people. Otherwise, the innovation will be considered like a new bureaucracy and rapidly discarded.
- Improvement is incremental and driven both by planning and exploiting opportunities.
  It is difficult to concurrently start a significant improvement of many processes. Usually the focus is on one or two processes at a time. Moreover, the improvement process is driven by a plan of actions, but in many cases, it may take advantage of not planned opportunities. For example, the need to deliver a complex project may be exploited for improving the delivery process of all applications.
- Both development and service view must be considered.
  An IT department does not produce software, but delivers services. This means that the system view is mandatory. The service level depends on the application software development, evolution and maintenance as well as the management of technological infrastructure and user support.

Our final consideration is that, even if process assessment and improvement concepts are less used in IT departments than in software development companies, they may play a significant role for improving the value delivered to the company business. The experience above described may be a good example and may provide useful suggestions.

The improvement process is still going on. The new main goals involve planning process and knowledge management for supporting the evolution of the in service applications.

7 References

[Fuggetta 2000] Fuggetta A., 2000, Software Process: A Roadmap. The Future of Software Engineering, Anthony Finkelstein (Ed.), ACM Press

[ISO15504] ISO/IEC 15504-1-9 Information technology-Software process assessment-Part 1-9

[ISO12207] ISO/IEC 12207 Information technology-Software life cycle processes

[Bucci 2000] Bucci G., Campanai M., Cignoni G.A., 2000, Rapid Assessment to solicit Process Improvement in SMEs, EuroSPI2000

[Calvo-Manzano 2002] Calvo-Manzano Villalón J.A., Cuevas Agustín G., San Feliu Gilabert T. Experiences in the Application of Software Process Improvement in SMES. Software Quality Journal, 10, 261–273, 2002

[SPIRE 1998] The SPIRE Project Team, 1998. The SPIRE Handbook, Marty Sanders (ed)

[Lamprecht 1998] Lamprecht W., Messer T., 1998. Software Process at the Gate to the Top. EuroSPI98

[Lindholm 1998] Lindholm T., 1998, Implementing SPI: Combining business process improvement and system development at Nokia, EuroSPI98

[Mc Caffery 2008] Mc Caffery F., Richardson I. and Moller P. Automotive–Adept: A Lightweight Assessment Method for the Automotive Software Industry. Softw. Process Improve. Pract. 2008; 13: 345–353

[Armbrust 2008] Armbrust O., Ebell J., Hammerschall U., Munch J. and Thoma D. Experiences and Results from Tailoring and Deploying a Large Process Standard in a Company. Softw. Process Improve. Pract. 2008; 13: 301–309

[Downey 2008] Downey F. and Coleman G. Using SPI to Achieve Delivery Objectives in e-Commerce Software Development. Softw. Process Improve. Pract. 2008; 13: 327–333

[Sommerville 2005] Sommerville J., Ransom J. An Empirical Study of Industrial Requirements Engineering Process Assessment and Improvement. ACM Trans. on Software Eng. and Methodology, Vol. 14, No. 1, January 2005, pp 85–117.

[Salvaneschi 2006] Salvaneschi P., Grasso D., Besurga M. The role of process improvement in IT departments: experiences and lessons learned. Software Process: Improvement and Practice (SPIP), John Wiley & Sons. June 2006, Volume 11, Issue 3, Pages 309 - 317

8 Author CVs

Paolo Salvaneschi
Associate professor of Software Engineering at University of Bergamo, Faculty of Engineering and owner of “Salvaneschi & Partners” (Software Engineering consulting company). He joined the University after spending twenty-five years as software architect and head of software development groups in industry and research labs. His professional experience is in software engineering as well as ICT applications in various areas (industry, civil engineering, environmental control). He published some hundred papers.

Daniele Grasso
Software Division manager at Mediamarket S.p.A IT Department

Maurizio Besurga
He has been IT Manager for Volvo Truck Company in various European Countries and is currently Chief Information Officier for Mediamarket S.p.A.

Service Level Management for IT Services in small settings: a Systematic Review

Calvo-Manzano Jose[1], Cuevas Gonzalo[1], Gómez Gerzon[2], Mejía Jezreel[1], Muñoz Mirna[1], Rabbi Forhad[1], San Feliu Tomás[1]

[1]Universidad Politécnica de Madrid, Facultad de Informática
{joseantonio.calvomanzano, gonzalo.cuevas, ma.munoz, tomas.sanfeliu}@upm.es, , jmejia@mpsei.fi.upm.es, i.am.rabbi@gmail.com
[2]Universidad Autónoma de Tamaulipas, Unidad Reynosa Rodhe
{ggomez@uat.edu.mx}

Abstract

This work presents the application of a systematic review protocol for Software Engineering. This protocol is used as a formal model by applying systematic review to Service Level Management. The objective is to search for papers related to Service Level Management for IT Services in small settings (including small and medium enterprises). Results obtained show that Service Level Management area is an increasing research field and there is a need for more in-depth studies.

Keywords

SLM, ITSM, Small Settings, Systematic Review, IT Services.

1 Introduction

Nowadays, most countries have transitioned from agriculture and manufacturing economies to service based economies. More than 75% of the economies of industrialized nations are based on business services [1]. According to the document “Capability Maturity Model Integration for Service (CMMI-SVC) Overview” published by Software Engineering Institute [2], demand for process improvement in services is likely to grow: services constitute more than 80% of the US and global economy. According to Gartner [3], 80% of the cost of an infrastructure is in service delivery and service support. Therefore, services are gaining significant importance day by day in industry.

Hence, industries need some sort of framework, standards to manage their services, specially the IT Services, because the development of IT has had a greater impact in the processes of businesses in the last years [9]. For this reason, IT Service Management (ITSM) has been developed. ITSM is a discipline for managing information technology systems, philosophically centered on the customer's perspective of IT's contribution to the business. The IT infrastructure needs to be aligned to the business requirements such that business unit and IT operate in a coordinated effort to achieve the goal of the organization. There are different frameworks for ITSM. Standards, process and evaluation models or framework help organizations to improve the management of their services projects. Some public framework and standards assessed relevant to service management are: ISO/IEC 20000 [13], Information Technology Infrastructure library (ITIL) [5], Capability Maturity Model Integration for Services (CMMI-SVC) [2], Control Objectives for Information and related Technology (COBIT) [8], eSourcing Capability Model for Service Providers (eSCM-SP) [4].

The previous models consider the IT Services from large companies’ perspective. But the fact is, nowadays small companies are gaining more importance. In Spain, according to Central Business Directory (DIRCE) [14], most companies are categorized as Small and Medium Enterprises (SMEs). In January of 2007, 99,81% of businesses, representing three million enterprises were small or medium. This statistic shows the importance that SMEs have in microeconomics (see Table 1).

Due to previous issues, a growing number of organizations are focusing on Service Level Management (SLM) process to determinate the level of IT Services that is needed to support the services offered. SLM provides an approach combining process management and industry best practices to ensure that the required and cost justifiable service quality is maintained and gradually improved [5]. As a result, SLM becomes a key concern nowadays.

This work addresses SLM process, from the point of view of small settings (include SMEs). Section 2 shows the importance of Service Level Management for IT Services. Section 3 and 4 describes the Systematic Review Method applied in this research work. Section 5 shows the results obtained. Finally, section 6 establishes a brief summary.

2 Importance of Service Level Management for IT Services

SLM is a vital process for every IT service provider organization in that it is responsible for agreeing and documenting service level targets and responsibilities within Service Level Agreements (SLAs) and Service Level Requirements (SLRs), for every activity within IT [6].

Moreover one of the important processes to regulate the qualities and to decrease the cost of IT services is the Service Level Management [15]. Also the ITSM models, standards and proposals give high importance to Service Level Management in the IT Service Management context.

The main factors to carry out this research focused in a systematic review for SLM from the point of view of small settings are:

- The growing current trend to acquire Technology and IT Services derivates of this acquisition by organizations [16].

- The importance of IT Service Management in small settings.
- The absence of models that help to implement the Service Level Management process in the context of the IT Service Management for Small Settings.

The statistics compiled by the National Statistical Institute (INE) and the Central Business Directory [8], show that small companies represent the highest percentage. The data are linked to factors such as income and sectorial concentration of such companies. The INE and DIRCE analyze their situation within the European Union and their relationship with the employees they recruit.

Table 1 summarizes the study for Spanish companies based on the employee stratum and total percentage [8]. The interest in showing this study is due to the current importance that small companies have.

Micro-enterprise	Small	Medium	Large	Total	SMEs
3,137.46	169.60	23.52	6.07	3,336.66	3,330.58
94,03%	5,08%	0,70%	0,18%	100%	99,818%

Table 1. Spanish Companies (employee's stratum and total percentage).

Systematic review gives a summary of the state of the art for this specific topic. In this case for IT Services Management in small settings. In addition systematic review identifies the existing gap in some topics related with SLM in SMEs. This results can help users who are working with SLM or can help to know current initiatives in this domain.

3 Description of Systematic Review Method

3.1 Systematic Review Background

Integration of research results was introduced for the first time in twenty century. In 1904, Pearson calculated the average of correlations between the typhoid fever inoculation and morality. Then, systematic review began to be formalized and at the end of the 80's systematic review achieves legitimacy as a field of research [10]. Later, Kitchenham [11] evolutes the idea of Evidence-Based Software Engineering and proposes a guideline for systematic reviews that is appropriate for software engineering researchers.

3.2 What is a Systematic Review

Systematic Review (SR) is used to refer to a specific methodology of research, developed in order to gather and evaluate the available evidence pertaining to a focused topic [10]. This is the process of summarize all existing information about a phenomena in thorough and empirical way. At the end, systematic review draws a general conclusion from individual studies on any phenomena.

A systematic type of review follows a very well defined and strict sequence of methodological steps. A systematic review begins when researchers are confident that it is necessary to carry it out. It aims to integrate empirical research in order to create generalizations. In this regard, defined assessment objectives, reference source, data extraction method are some of the aspects contained in the protocol used for this systematic review [10].

3.3 Protocol Description

Biolchini et al [10] have drawn up a proposal for how to conduct a systematic review focused on Software Engineering adapting it from other study area such as medicine.

Hence, for this work, the protocol proposed is applied to the “Service Level Management for IT Services in small settings: a Systematic Review”.

4 Prototype Development

Next, the prototype development used for the systematic review of the subject is presented: Service Level management for IT Services in small settings.

4.1 Question Formulation

The systematic review objective should be clearly established in order to formalize the question:

4.1.1 *Question Focus*

The systematic review is carried out to identify initiatives and experience reports on Service Level Management for IT Services in small settings.

4.1.2 Question Quality and Amplitude

This section aims at defining the syntax of the research question (the context in which the review is applied and the question the study must answer) and its semantic specificity (or question range) described by the remaining items of this section - intervention, effect, outcome mesasure, population and application. Next, each of them are described [10] specifically for Service Level Management for IT Services in small settings.

- Problem: Service Level Management (SLM) is a vital process for every IT service provider organization in that it is responsible for agreeing and documenting service level targets and responsibilities within SLAs and SLRs, for every activity within IT. SLM implementation is needed to ensure that an agreed level of IT service is provided for all current IT services, and that future services are delivered to agreed achievable targets [5].
- Question: What initiatives have been carried out to evaluate processes for Service Level Management in IT service context?
- Intervention: Service Level Management for IT Services in small settings.
- Effect: Service Level Management initiatives and proposals for IT services in small settings.
- Outcome measure: Number of identified initiatives.
- Population: Publications related to Service Level Management, IT Services and small settings
- Application: Organizations that use IT services and those who provide them.
- Experimental Design: None experimental design will be performed.

4.2 Source Selection

The objective of this section is to select the sources where the primary studies will be executed [10]. To perform the selection the author of the systematic review protocol proposes to address the following issues:

4.2.1 Source Selection Criteria Definition

- Use search mechanism with keywords and sites suggested by experts.
- Use papers recommended by other experts.
- Use papers available on the website.

4.2.2 Studies Language

- English.

4.2.3 Source Identification

Sources Search. The identification of sources has been based on the criterion of experts in our research area. These sources include journals as: European Journal of Operational Research, Information and Software Technology, Software: Practice and Experience, Software Process: Improvement and Practice, IEEE Software, Software Technology and Engineering Practice, Computer and research workshops & technical reports of Software Engineering Institute – SEI, among others.

Search Strings. Keywords from the word set defined in the question were extracted. Combining these keywords with the logical operators “AND” and “OR”, two search strings were obtained (see Table 2). These search strings have been adapted for each web browser of the sources.

	Search String
1	itsm or slm and sla and ((itil or asl or 20000 or 15000 or cmmi-svc or mof or itscmm) and (small and (company or organization or enterprise or setting)))
2	‘it service management‘ and 'service level management' and (small and (company or organization or enterprise or setting))

Table 2. Search strings

Source List. These sources have been selected taking into account the defined source search method (see Table 3).

#	**Source**
1	ACM Digital Library
2	IEEE Computer Science Digital Library
3	Springer Link
4	Science@Direct
5	Software Engineering Institute

Table 3. Search strings

4.2.4 Source Selection After Assessment:

First, it was evaluated if the sources fit all defined criteria. Initially, the complete list is right. After applying the search string to all sources, it was found that some items were common in the IEEE Computer Science Digital Library and ACM sources.

4.2.5 Reference Checking

Three researchers from the Research Group of Software Process Improvement for Spain and Latin American Region evaluated the sources list obtained from the previous section and determined, at first instance, all references as approved.

4.3 Studies Selection

In this systematic review an iterative and incremental procedure is used for studies selection: a) Iterative, to group all activities that could be repeated during the procedure, and b) Incremental, because the studies are approached and recorded one by one until obtaining the systematic review results [12]. This iterative and incremental procedure is used due to its functionality in other systematic reviews. This section describes the process and criteria for studies selection and evaluation.

4.3.1 Studies Definition

The studies inclusion (IC) and exclusion criteria (EC) definitions [11] are as follows:

Acronym	Criteria Description
IC1	Include papers whose title is related to Service Level Management and IT Services and small settings
IC2	Include papers that contains keywords that match with those define in the search string
IC3	Include papers whose abstract is related to the topic considered
IC4	Include papers after partial or total reading
EC1	Exclude those papers that do not match with the previous inclusion criteria
EC2	Exclude all duplicate papers

Table 4. Studies inclusion and exclusion criteria definition

Studies Types Definition. Initially all studies related to Service Level Management will be taken into account. However, the greatest interest will focus on studies that show results on Service Level Management for IT services in small settings.

Procedures for studies selection. With regards to the selection criteria, the title was initially the main criterion; nevertheless, in some cases it did not provide enough information, thereby reading the summary of each of them was necessary and in some cases a review of the full text was required.

4.3.2 Selection Execution

- Initial Studies Selection. At first a search execution was conducted to verify the parameters used by each engine and adapt search string to them. Table 5 shows in the column “Found” the obtained value.
- Study Quality Evaluation. To determine the quality of the study, some participants of the research group, applying IC and EC, obtained the primary studies (see Table 5).

Sources	Search date	Found	Primary studies
IEEE	05/11/09	59	11
ACM	05/11/09	12	3
Springer	05/12/09	59	12
Science@Direct	05/13/09	42	3
SEI	05/14/09	10	3
Total		182	31

Table 5. Number of studies and results derived from each source

Next, a quality study is evaluated to obtain the assessment results that permit us to quantify those studies that effectively support the stated objectives.

The previous information constitutes the study basis for following the systematic review process, and checking the quality of the study.

4.4 Information Extraction

This section begins once primary studies are selected. Then, in this section, extraction criteria and result are described

4.4.1 Information Inclusion (ICinf) and Exclusion (ECinf) Criteria Definition:

Acronym	Criteria Description
IC1inf	Collect Information about the organization's trend respect to service level management.
IC2inf	Classify processes followed by companies for IT service management.
IC3inf	Identify proposed methodologies, methods and procedures in studies for IT service management
EC1inf	Exclusion the information that is not related to the inclusion criteria defined above.

Table 6. Information Inclusion and Exclusion Criteria Definition

4.4.2 Data Extraction Forms

To analyze data and information submitted in selected studies, relevant remarks of the main studies ideas were made and recorded in a document with a sequence number identification that matches with the sequential paper number given when it was stored a primary study.

4.4.3 Extraction Execution

Objective Results Extraction: A complete and detailed reading from these studies allowed us to organize and classify then for a later analysis. With an unbiased evaluation of the information, identified and classified studies records were generated in a structured table containing the following rows: Consecutive Study (sequential paper number), Study methodology (remarks of the main ideas concerned with the methodology), and Study outcome (data and information of the conclusion presented in each study).

Subjective Result Extraction: The following rows where added to the previous table: Data about Authors (full names and available contact information in the studies) and Additional Notes (a specific field to store general information related to subject covered in the study).

4.4.4 Resolution of divergences between reviewers

In the development procedure for the extraction of the information contained in the selected primary studies, different perceptions among authors of the studies were presented. However, none of them is considered as an important divergence, but rather, the findings were complemented to obtain a comprehensive analysis of the paper.

5 Result Summary

After the systematic review execution, the results must be summarized and analyzed using the statistical methods defined during the planning phase. This section presents a summary of the data resulting from the selected studies. This summary is obtained from statistical calculus.

5.1 Studies Trends

In order to know organization trend respect to Service Level Management (SLM), studies related to any aspect of SLM were classified, taking into account that "SLM" term refers to both Service Level Agreement (SLA) and Service Level Operation (SLO) terms.

Figure 1 shows two types of trends. 1) Between 2002 and 2006, there is a polynomial trend of order 3 because data fluctuate along the graphic. And, B) from 2006 to 2008 data have a linear trend because the studies are increasing at a constant rate. The trend shows from year 2006 the increasing interest related to Service Level Management.

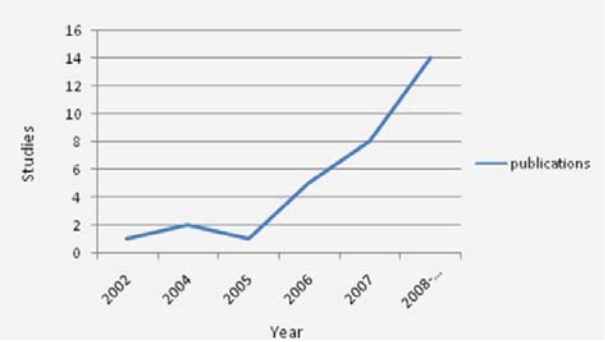


Fig. 1. Studies Trend

5.2 Studies Classification

During the protocol development, and using the studies selection's methodology and information analysis found in each study, it was possible to determine that studies could be classified into four items.

Those studies covering:

A) topics related only to ITSM and SLM.
B) topics related only to ITSM and small settings.
C) a relationship of three topics (ITSM and SLM and small settings), and

D) other studies that have no relationship with these issues (ITSM and SLM and small settings).

Figure 2 shows the studies percentage for each item according to the previous classification.

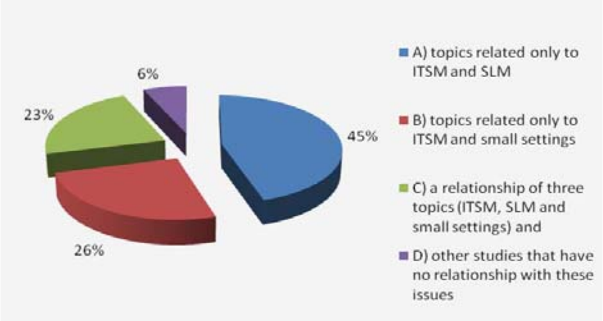


Fig. 2. Studies Classification

Figure 2 shows that the 45% of the primary studies (see Table 5) are related to ITSM and SLM what confirms the trend shown in Figure 1. The 26% shows information related to ITSM and small settings. However, it is necessary to highlight that only 23% is related to item C) that grouped the three topics (ITSM, SLM and small settings). Finally a 6% has been eliminated by the exclusion criteria EC1inf.

5.3 Analysis of those studies covering the relationship of three topics (ITSM, SLM and small settings)

Figure 3 shows the results of relationship related to topics (ITSM, SLM and small settings). In the analysis, it is noted that 23% of the papers comprise three aspects (basis of the ongoing systematic review). From here all analyses are referred to this 23% (item C).

5.3.1 Analysis by country

The source of 23% of the papers is shown in Figure 3. United States of America is one of the main countries involved in the study of ITSM, SLM and small settings, with 29%. The remaining studies are divided in the countries South Korea, Sweden, French, Canada and Germany with 14% respectively.

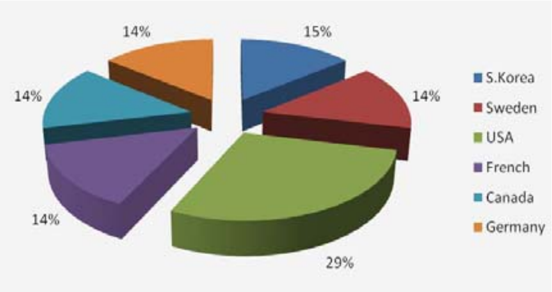


Fig. 3. Studies Classified by country

5.3.2 Analysis by company size

Figure 4 shows information about companies' size. Due to most of the studies are short of information about companies, Figure 4 has been made taking into account two criterions: 1) the application of studies in the companies' size, 2) the studies did not mention the companies' size.

Hence, according to Figure 4, most of the companies that support their research are small companies, 85,7%. Remaining 14,3% companies are not mentioned.

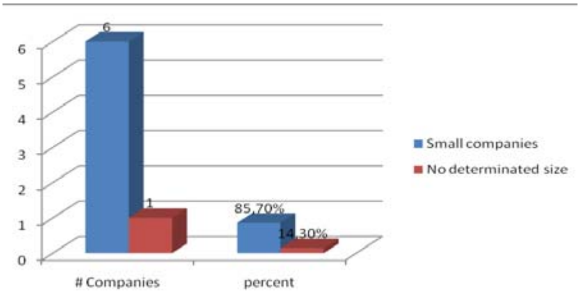


Fig. 4. Studies by companies size

5.3.3 Analysis by models

With respect to item C only the 78% have used ITIL. COBIT and ISO 20000.

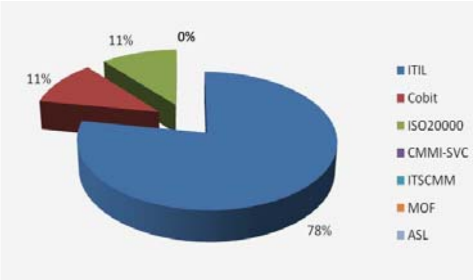


Fig. 5. Studies Classified by models

6 Conclusions

This work shows the results for Systematic Review of the studies related to Service Level Management in small settings. It was obtained using the protocol proposed by Biolchini et al [10].

The obtained results show the status of the art for "Service Level Management in small settings". We can see most of the works have been performed in year 2008. According to the statistics we find from year 2005 the works on Service Level Management are increasing significantly. This trend confirms the interest over SLM process showed by Gartner and CMMI-SVC reports.

Other relevant data showed in this work is that 45 % of studies papers deal with IT service management in terms of SLM (include SLA and SLO). The 26% of the studies is related to small setting (include SMEs). Moreover, 23 % of studies are related to SLM and small settings.

The studies show that ITIL, COBIT, and ISO2000 are the most important models in a Service Level management context.

These results can help users who are working with SLM or can help to know current initiatives in this domain.

7 Acknowledgments

This work is sponsored by Endesa, everis Foundation through the Research Group of Software Process Improvement for Spain and Latin American Region, as well as by the Secretariat of Public Education (Mexico) with a scholarship PROMEP through the agreement with the Autonomous University of Tamaulipas.

Appendix A: List of primary studies in the systematic review

Next are presented the selected primary studies in the development of this systematic review.

[1] NeilMcBride, "Exploring service issues within the IT organization: Four mini-case studies", International Journal of Information Management, 2009

[2] Stefan Thanheiser, Lei Liu, Hartmut Schmeck: "SimSOA: an approach for agent-based simulation and design-time assessment of SOC-based IT systems", Proceedings of the 2009 ACM Symposium on Applied Computing (SAC), Honolulu, Hawaii, USA, March 9-12, 2009

[3] CMMI Product Team, "CMMI® for Services, Version 1.2", Technical Report CMU/SEI-2009-TR-001, Software Engineering Institute, USA, March, 2009

[4] Mira Kajko-Mattsson and Christos Makridis, "Evaluating SLA Management Process Model within Four Companies", The Third International Conference on Software Engineering Advances, Sliema, Malta, 2008

[5] Thomas Schaaf, Michael Brenner, "On Tool Support for Service Level Management: From Requirements to System Specifications", Proceedings of BDIM 2008, 3rd IEEE/IFIP International Workshop on Business-Driven IT Management, , Salvador, Brazil. IEEE 2008

[6] Genady Grabarnik, Heiko Ludwig, Larisa Shwartz, "Dynamic Management of Outsourced Service Processes' QoS in a Service Provider – Service Supplier Environment", International Workshop on Business-Driven IT Management, BDIM 2008: Salvador, Brazil, 2008

[7] Adrian Paschke, Martin Bichler, "Knowledge representation concepts for automated SLA management", The Computing Research Repository (CoRR), Decision Support Systems 46 (2008) 187–205, 2008

[8] Jung-Oh Park, Sang-Geun Kim, Byeong-Hun Choi, Moon-Seog Jun, "The study on the maturity measurement method of security management for ITSM", International Conference on Convergence and Hybrid Information Technology, Gyeongbuk, S. Korea, 2008

[9] Rajeev Gupta, K Hima Prasad, Mukesh Mohania, "Automating ITSM Incident Management Process", International Conference on Automonic Computing, Chicago, IL, USA , 2008.

[10] Haining Wang Shouqian Sun Yanan Huang Shiwei Cheng, "An ITIL-Based IT Service Management Model for Garment Enterprises", International Conference on Information Management, Innovation Management and Industrial Engineering, 2008. ICIII '08, Taipei, Taiwan, 2008

[11] Nabiollahi, Akbar bin Sahibuddin, Shamsul , "Considering service strategy in ITIL V3 as a framework for IT Governance, International Symposium on Information Technology", ITSim 2008, 26-28 Aug. Kuala Lumpur, Malaysia, 2008

[12] Jin, K. Ray, P. "Business-Oriented Development Methodology for IT Service Management", Proceedings of the 41st Annual Hawaii International Conference on System Sciences, Hawaii, USA, 2008

[13] Philip Bianco, Grace A. Lewis, Paulo Merson, "Service Level Agreements in Service-Oriented Architecture Environments", Technical Note CMU/SEI-2008-TN-021, Software Engineering Institute, 2008

[14] Michael Brenner, Gabi Dreo Rodosek, et al, "Provisioning: Challenges, Process Alignment and Tool Support". Handbook of Network and System Administration, 2008, Pages 855-904

[15] Larisa Shwartz, Naga Ayachitula et al, IT Service Provider's Multi-Customer and Multi-Tenant Environments, The 9th IEEE International Conference on E-Commerce Technology and The 4th IEEE International Conference on Enterprise Computing, E-Commerce and E-Services(CEC-EEE 2007), Tokyo, Japan,2007

[16] Christian Braun, Robert Winter, “Integration of IT service management into enterprise architecture”, Symposium on Applied Computing archive, Proceedings of the 2007 ACM symposium on Applied computing table of contents, Seoul, Korea, 2007

[17] Jiangping Wan Yunfeng Wang, Chuwei Zheng, “Research on IT Service Management Knowledge Support Structure”, International Conference on Wireless Communications, Networking and Mobile Computing, WiCom 2007, Shanghai, China, 2007

[18] Larisa Shwartz, Naga Ayachitula, Melissa Buco, Maheswaran Surendra, Christopher Ward, “Service Provider Considerations for IT Service Management”, 10th IFIP/IEEE International Symposium on Integrated Network Management, Munich, Germany, 21-25 May 2007

[19] Rob Addy, “Effective IT Service Management”, ISBN 978-3-540-73197-9 Springer Berlin Heidelberg New York, 2007

[20] Halina Kaminski, Mark Perry, “Employing Intelligent Agents to Automate SLA Creation”, Whitestein Series in Software Agent Technologies and Autonomic Computing Book Emerging Web Services Technology, 2007

[21] Brenner M., Dreo Rodosek G., Hanemann A., Hegering H.-G., König R. “Service Provisioning: Challenges, Process Alignment and Tool Support”. Handbook of Network and System Administration, J. Bergstra, M.Burgess (editors) Elsevier, 2007, ISBN 978-0-444-52198-9.

[22] Hao Wang Bo Yang Liang Liu Qian Ma Ke Wei Sun Ying Chen , “Knowledge Enhanced IT Service Management”, IEEE International Conference on e-Business Engineering, ICEBE 2007, Hong Kong, 2007.

[23] Sven Graupner, Nigel Cook, Derek Coleman, Tilo Nitzsche, “Platform for Delivering IT Management Services”, Hewlett-Packard Laboratories, 1501 Page Mill Road, Palo Alto, CA 94304, USA, 2006

[24]Software Engineering Institute, “Improving Processes in Small Settings (IPSS) A White Paper”, The International Process Research Consortium (IPRC) Pittsburgh, PA 15213, 2006

[25] Michael Brenner , “Classifying ITIL Processes: A Taxonomy under Tool Support Aspects”, IEEE/IFIP International Workshop on Business-Driven IT Management (BDIM 2006) Vancouver, Canada, April 7, 2006

[26] Rodrigo Bonfá Drago, Rodolfo da Silva Villaça, “A Proposed Web Tool for IT Environment Service Management”, Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW 2006), Guadeloupe, French Caribbean, 2006

[27] Akhil Sahai and Sven Graupner, “Enterprise Management and Web Services,Web Services in the Enterprise”, Springer US ISBN 978-0-387-23374-1 (Print) 978-0-387-27597-0 (Online), 2005

[28] Thorsten Anders, “Development of a generic IT service catalog as pre-arrangement for Service Level Agreements”, 10th IEEE Conference on Emerging Technologies and Factory Automation, 2005. ETFA, Catania, Italy, 2005.

[29] Jos J.M. Trienekens, Jacques J. Bouman, Mark van der Zwan, “Specification of Service Level Agreements: Problems”, Principles and Practices, Software Quality Journal, Volume 12, Number 1 / March, 2004, Springer Netherlands.

[30] Claudio Bartolini and Mathias Sallé, “Business Driven Prioritization of Service Incidents”, 15th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management, DSOM 2004, Davis, CA, USA, DSOM 2004, LNCS 3278, pp. 64–75, 2004.

[31] V.P. Wade, “Service management and the telecommunications information networking”, Computer Communications 22 (1999) 1633–1637, Elsevier Science B.V. Dublin, Ireland.

8 Literature

[1] National Academy of Engineering. The impact of Academic Research on Industrial Performance. The National Academic Press, Washington, DC, 2003.

[2] Eileen Forrester, CMMI for Service (CMMI-SVC) Overview, Software Engineering Institute, Carnegie Mellon University, USA , 2006

[3] Lynch, C. G. “Most Companies Adopting ITIL® Practices,” CIO Magazine, March 1, 2006.

[4] Stuart, G., Ronald, D., Jim, J.Q, Sue C, Information Technology Service Management-An Emerging Area for Academic Research and Pedagogical Development, SIGMIS-CPR’07, St.Louis, Missouri, USA, 2007

[5] Office of Government Commerce, ITIL: Service Design, The Stationary Office. 2007

[6] Office of Government Commerce,The official introduction to the ITIL service lifecycle: the official introduction to the ITIL service lifecycle, The Stationary Office, 2007

[7] Hochstein, A., Zarnekow, R., Brenner, W.: ITIL as Common Practice Reference Model for IT Service Management – Formal assessment and implications for practice, in: Proceedings of the 2005 IEEE International Conference on E- Technology, E-Commerce and E-Service, Hong Kong, 2005.

[8] Directorio Central de Empresa DIRCE, Empresas españolas (estrato de asalariados y porcentaje del total) DIRCE 2007. http://www.ine.es/inebase/cgi/axi

[9] http://www.interpromusa.com/downloads/Book%20Content/Introduction%20to%20ITIL.pdf, February 4, 2009

[10] Biolchini J.,Gomes M.P., Cruz N.A., Horta T.G., "Systematic Review in Software Engineering", RT ES679/05, 2005

[11] Kitchenham, B. A., Dyba, T. and Jorgensen, M., Evidence-based Software Engineering, 26th International Conference on Software Engineering (ICSE 2004), Scotland, 2004.

[12] Pino J.F., G.F., Piattini M., *Software process improvement in samll and medium software enterprises: a systematic review.* Softwaer Quality Springer, 2008. **16**(2): p. 237-261.

[13] ISO/IEC-20000-1 (2005). Information Technology- Service Management-part 1:Specification, International Standard Organization.

[14] DIRCE, D.C.d.E. Empresas españolas (estrato de asalariados y porcentaje del total) DIRCE 2007.

[15] Johnston, R.B.a.R., Service Agreements- A management Guide. first ed. ITSM Library, ed. J.V.B.a.A.d. Jong. 2006: Van Haren Publishing.

[16] AEC (2007), "La consultoría en España", Asociación Española de Empresas de Consultoría. El sector en cifras 2007

9 Author CVs

Calvo-Manzano Jose A.

He is PhD in Computer Science. He is assistant professor in the Computer Science School at the Polytechnic University of Madrid. He is teaching in the area of Software Engineering, specifically in the domain of software process management and improvement. He has participated in more than 20 research projects (European and Spanish Public Administration). He is author of more than 50 international papers. He is author of books related to software process improvement and software engineering topics also. He is a member of SOMEPRO research group, where his main research field is process improvement, focusing in the topics of requirements management, project management, process deployment, and solicitation and supplier agreement development.

Cuevas Gonzalo

He received an Engineering degree in Telecommunications in 1965 and a PhD in Telecommunications in 1974. He also received an MS in Computer Science from the Polytechnic University of Madrid in 1972. He has been vice dean of the Computer Science faculty at the Polytechnic University of Madrid, where he is a full professor since 1970. He worked for Iberia Airlines as Software Development Director, supervising over 200 technicians, Data Processing Centre Director, Data Transmission Software Development Director, and the person in charge of strategic planning and software process improvement. He led European projects on software best practices from 1991 to 1995. His main research field is software engineering, including both technology (methods, techniques, and formalisms) and management. His current research interest is process models and methods, and transition packages. He is member of ACM, senior member of IEEE, member of the Telecommunication Engineering Association and member of the Computer Sciences Association.

Gómez Gerzon

He is a B.Sc. in Computer Science, Autonomous University of Tamaulipas (UAT), Mexico. He was assistant professor in the same university. He has occupied many management positions during more 4 years in the industry. He has published several technical papers on IT Service Management. Nowadays, he is a PhD student in the Faculty of Computer Science of the Polytechnic University of Madrid, Spain. He works in a “Metamodel for Service Level Management”.

Mejia Jezreel

He is a Master in Computer Science (2003-2005) and Computer Science Engineer (1996-2000) from the Institute Technological of Orizaba University of México. He is actually Software Engineering PhD student from the Computer Science faculty at the Polytechnic University of Madrid. His research field is Software Engineering. His current research interest consists on outsourcing process improvement. He has published several technical papers on acquisition process improvement. He is colaboring with an everis Foundation research project.

Muñoz Mirna

She is a Master in Computer Science (2002-2004) and Computer Science Engineer (1996-2000) from the Institute Technological of Orizaba University of México. She is actually Software Engineering PhD student from the Computer Science faculty at the Polytechnic University of Madrid. Her research field is Software Engineering. Her current research interest consists of the development of a methodology whose goal is to incorporate process improvement focusing on diminishing change resistance. She has published several technical papers on project management and process improvement. She is colaboring with an everis consultants research project.

Rabbi Forhad

He is a B.Sc. in Computer Science and Information Technology, Islamic University of Technology (IUT), Bangladesh. He is a lecturer at Shah Jalal University of Science and Technology (SUST), Bangladesh. Nowadays, he is attending dual master program on software engineering under Faculty of Computer Science of the Polytechnic University of Madrid, Spain and Blekinge Institute of Technology. He is working on “Elaborating IT Service Taxonomy and Related Patterns in Service Level Management”

San Feliu Tomás

He is PhD in Computer Science. He has been working 15 years to Software Engineering field as programmer, associate and assistant professor and consultant. He has been involved in CMM appraisals in Spain and Latin America. As Software Engineering assistant professor at the Polytechnic University of Madrid, his main research field is software process and software process improvement. His current research interest includes project management and risk management. He has published several books on process improvement and he has published technical papers in software engineering and software process improvement.

Integration of CASE Tools to Software Processes: A Case Study

K. Alpay Erturkmen and Onur Demirors
METU Informatics Institute
Ankara, Turkey
alpaye@gmail.com, demirors@metu.edu.tr

Abstract

In this paper we present the results of a case-study we have performed in a software development organization to understand the extent of CASE tool use in software development, observe both the CASE tool integration problem and the complexity of user interactions with these tools in practice. The study was performed to further analyze the needs and refine the details of our research, based on our hypothesis that partial process automation can be achieved with the support of an integrated set of CASE tools that can execute modelled processes on behalf of users. We have analyzed process definitions of a software organization to identify CASE tools, user interactions with these tools and structured portions of process definitions. The result of the case study showed the existence of extensive CASE tool interactions in software development, a number of problems for CASE tool integrations and partially structured process definitions interacting with CASE tools.

Keywords

Software Development Process Automation, CASE Tool Integration, Case Study, Process Modelling

1 Introduction

Modeling is often required for successful implementations of business process re-engineering (BPR) and automated business process support in enterprises [1,2]. BPR involves the redesign of an organization's business processes. When “process automation” is necessary as part of BPR initiatives, process modeling becomes a vital task in re-designing work and allocating responsibilities between humans and computers.

For software development processes a multitude of CASE (Computer-Aided Software Engineering) tools are employed. These tools provide sophisticated islands of functionality without ample integration facilities. Thus they can only present software automation capabilities for specific functionality. Existing integrations are maintained by tool vendors and are almost always point-to-point, resulting in reduced functionality, integration related problems and vendor dependency.

An issue which further complicates process automation is related with the nature of software development. Software development, especially “new” software development is almost completely a creative, non-structured effort [3]. These processes require flexible flows and decision making ability from the actors. This is considered as one of the reasons why automation of software development has not been successful for wide audiences.

We hypothesize that partial process automation can be achieved with the support of an integrated set of CASE tools that can execute modeled processes on behalf of users. In addition to establishing a baseline for integration, the approach will unify the two separate process modeling based activities for process improvement and process automation.

The complexity of software development has significantly risen as software became a core asset for organizations. It can be quite difficult to analyze these processes as a whole due to complex organizational phenomena involved. Modification of the environment with the introduction of a new method or technology can be costly, time consuming and can have unforeseen consequences. Even more, both employees and management may resist change without solid proof for improvement.

Developing experiments that simulate the operation of such processes can also be infeasible because of this complexity. Experiment designs become complex since they have to take into account a number of critical relationships. It is very hard to find theoretical basis for problems involving CASE-tool usage and their integrations since they cut across specialized domains of software engineering. They are also mostly vendor-dependent. To explore the status and issues with CASE tool usage and their integrations under these constraints and investigate the potential of our hypothesis, we have designed and performed a case-study.

We have analyzed the process definitions of a software development organization. The software development processes defined by the organization have recently been assessed to be CMMI Maturity Level 3. The group is solely focused on software development and uses a large number of CASE tools including configuration, requirements, test management, automated functional testing, UML modeling, risk management, time tracking tools and IDEs. The case study work involved the analysis of process definitions written in natural language. The analysis aimed the identification of process model components including actors, actions, events, and interactions. The results enabled us to determine the extent and nature of CASE tool usage and the frequency of user interactions with CASE tools in the organization.

Our paper is organized as follows; Section 2 provides related research on process modeling and CASE tool integration while Section 3 details our case study design and case study plan. Section 4 presents the results of the case study, the discussions on these results are given in Section 5. Section 6 provides our conclusions and future directions.

2 Related Research

Process automation is a well-contributed research area. It is widely-practiced on industries like banking, insurance and production which have well-defined and structured processes. On the other hand, software development presents fundamental challenges to process automation. Our research to support our hypothesis for partial process automation in software development focuses mainly on two of these challenges. One of these challenges is the current state of the tools used in software development, while the other is the unstructured, creative nature of software development processes.

The tools employed by software development organizations are separated and specialized for each area of expertise. They offer integration mechanisms that are point-to-point, fragile, inflexible and vendor dependent [4]. The result is automation solutions that are either constrained to individual tools and processes or are complex, do-it-all vendor-dependent solutions. These factors prevent wider adoption of automation solutions [5]. Previous work in CASE tool integration frameworks, were grouped by [6] and [7] as follows:

- Standardization efforts or middleware services (CAIS [8], PCTE [9], CDIF [10], CORBA [11], RTP OTIF [12], agent based [13], etc.)
- Architecture models, infrastructures and tool suites (ECMA Toaster Model [14], ToolBus architecture [15] etc.)
- Basic tool integration mechanisms (data sharing, data linkage, data interchange, message passing, publish/subscribe services [16], [17])
- Integrated Process Support Environments (IPSE) like MARVEL, software development process automation through a rule-based architecture [18]
- Recent work on meta-model based tool integrations [6]

Although many integration frameworks have been proposed in the literature, none of them have been widely adopted in practice. Basic tool mechanisms does not support beyond basic requirements while models like PCTE are too complex to be feasible in practice. Recent advances in modeling domain rendered techniques like CDIF obsolete. The same is true for IPSEs with the wide-acceptance of specialized CASE tools [4]. Complete tool suites have become impractical; since they were unable to cover all development activities at the same time constraining the selection of best-of-class CASE tools.

Software development process automation through process centered software engineering environments has been addressed in the literature including [19], [18] and [20]. This is one of the first attempts to develop a software process automation environment using a tool-suite, named MARVEL. The approach taken was developing a rule-based IPSE supporting controlled automation. However the complexity and effort required for developing such a solution outweighed the benefits in practice [6].

[20] proposes the concepts “process integration” and “process driven CASE environment” over other CASE tool integration mechanisms. The paper describes an experiment on “Softman” environment where software process models are developed for this environment. The solution suggested in [20] considers a single development environment. Implementation for other development environments or a generic solution however would require extensive additional effort.

Recent model-based research [6] offers tool integration based on semantic technologies. Kapsammer’s work focuses on the development of vocabularies and ontologies for the software development domain. These ontologies are represented as meta-models to enable CASE tool inter-operation.

We have evaluated Eclipse ALF (Application Life Cycle Framework) [21] as a CASE tool integration framework. ALF is an Eclipse Foundation project that aims to provide an open integration solution for CASE tools. ALF provides a common vocabulary (ontology), standard interfaces and a controlling event manager. This framework helps organizations solve the integration problem without depending on specific vendors and tools. However, individual tools are required to support this initiative.

ALF uses a BPEL (Business Process Execution Language) [22] engine to execute complex processes triggered by events generated by different CASE tools. An example will be the execution of code analysis, security and unit tests on a new revision of code added in the version control repository. Here, creation of a new revision is the event that triggers a complex series of actions performed by different tools (possibly from different vendors). ALF integration infrastructure is well suited for automating the process definitions.

The other challenge presented by software development is the unstructured, creative nature of its processes. Such processes are difficult to handle using automation software like workflow-systems. These systems traditionally utilize well-defined, structured process definitions [23,24]. Process automation in unstructured environments has been addressed in recent research on Unified Activity Management [25,26,27,28]. The authors propose an ontology-based representation for unstructured work in organizations. The solution they provide is not to automate these processes but rather complement automation efforts.

3 Case Study

The case-study was designed so that its results will provide an understanding of the feasibility of our proposed future vision to the problems: partial process automation through an integrated set of CASE tools that can execute processes on behalf of users.

3.1 Design

There were two main questions posed by the case study:

- What is the extent and nature of CASE tool usage in software development?
- How frequently and in what manner do the users interact with CASE tools while executing software development processes?

These two questions were fundamental to answer more detailed questions like: How structured software development processes are? What is the complexity of user interventions in these structured processes? Is it possible to automate these processes? How multiple CASE tools interact and integrate with each other? How are these interactions and integrations managed?

To answer all these questions we have looked for the following information in process definitions:

- What is the fraction of process definitions containing interactions with CASE tools?
- Are there any sequences of non-creative (structured) actions within these process definitions involving multiple user interactions with (possibly different) CASE tools?
- What is the scope and complexity of the user interventions in these sequences?
- How can these user interventions be classified?
- Is it possible to remove these user interventions by delegating them to an automated entity (an “event manager”) for automation?

The questions we have developed posed some constraints on the selection of a target organization. The organization should be focused on software development. The development process should employ regular use of multiple CASE tools. An organization with defined processes (preferably certified in CMMI Maturity Level 3 or comparative ISO 15504 levels) is favored.

To provide answers to our questions, analysis of the software process definitions, supporting guidelines and related documents created by the organization was required. A preliminary scan of the process definitions was planned to filter out processes that have no CASE tool interactions and are non-structured, i.e. not suitable for automation [23].

The remaining processes would be further analyzed. The analysis would employ information gathering on those remaining processes to identify the process flow, events that trigger the actions, actors, CASE tools, related artifacts, extent of user interventions and context information. A tabular format was developed to keep the information for further analysis. For each process definition, sequences of actions, events and actors would be modeled using BPMN notation. Based on these process models, the interactions with the CASE tools would be identified and context information would be extracted. The analysis would be supported by interviews with process owners and key actors.

The outputs would be analyzed to identify the extent of CASE tool usage, the nature of the processes and interactions between users and CASE tools. If a process definition is found to have the following properties, than it is labeled to be suitable for automation:

- It involves multiple interactions with CASE tools
- The flow of the process does not include decisions. It does not require decision-making and creativity. It is a structured process.
- The interactions between users and CASE tools are simple and consist of simple actions. The interactions are focused over common (preferably single) artifacts.

These processes would provide a basis for the next phase of our study to identify sequences of actions in process definitions that are suitable for automation. They would also help us understand the requirements for our vision of an automation solution.

3.2 Constraints

The case-study design was constrained to a target organization with defined processes and an existing CASE-tool infrastructure. This choice helped us work on existing process definitions rather than the observation of actual execution. The same case study may provide a different outcome in an organization with undefined processes or with the observation of actual process executions. However these different results do not pose any threats to the validity of our case-study since defined processes compared to observed process definitions provide a consistent basis for analysis. In addition to this considering our vision, undefined processes are inherently harder or even impossible to automate [5].

Replicating this study on a group of organizations and comparisons of the results would provide more generalizable results. However the result of the case-study on a single organization would give us an understanding on the feasibility of our solution for this organization. The target organization we have defined for our case-study would provide a perfect opportunity for our solution.

3.3 Execution

The case study was performed in a software development branch of a research organization. This branch develops software and systems for military and civil foundations. To support their development efforts they utilize multiple CASE tools for requirements management, configuration management, change management, test management, automated functional testing, project planning, risk management and time tracking. We were provided access to the software development process definitions of the group which have recently been evaluated to be CMMI Level 3.

Process definitions were grouped similar to the process areas in the CMMI model [29]. We were guided by the process engineering support group and narrowed our work on process definitions for Project Management (PM), Configuration Management (CM), Requirements Engineering (RE), Technical Solution (TS) and Verification and Validation (VV). We excluded Process Management (PcM), Measurement and Analysis (MA), Software Quality Assurance (SQA), Risk Management (RkM), Organizational Training (OT) and Decision Analysis and Resolution (DAR) process areas. These processes are concerned with management and process improvement aspects of the operation, use at most one CASE tool (mostly software configuration management and change management) and are mostly unstructured (creative). We have extracted necessary information from only those process definitions that have CASE tool interactions and structured flow of actions. We then tabulated and modeled this information for further analysis.

The most prevalent difficulty we have experienced during our work was the level of detail given in process definitions. The process definitions were at times ambiguous or inconsistent in providing context, actor and event information. The missing information was obtained from process owners and actors through interviews or informal questions.

The case study was completed in 6 weeks from December 2008 to January 2009, including the preliminary scan and detailed analysis of process definitions. Analysis and interviews were performed on-site since access to process definitions was restricted. Secondary work like documentation, information organization and further analysis was performed off-site. Due to time constraints on both parties the duration was longer than originally planned to be around 2 weeks for a single full-time researcher.

Process Name	Actor	Action	Succeeded By	CASE Tools involved	Artifact/Data Assoication	Type of User Intervention
Example Process – Obtain commitment to software requirements	Team Leader	Get SRS template	Get SRS supplementary documents	SCM	Project.SRS.Template	Get artifact
	Team Leader	Get SRS supplementary documents	Generate SRS	SCM	Project.SRS.SupplementaryDocuments	Get artifact
	Team Leader	Generate SRS	Send SRS to Project Manager	RM	Project.SRS	Get artifact
	Team Leader	Send SRS to Project Manager	Get customer approval for SRS	E-mail?	Project.SRS	Submit artifact
	Project Manager	Get customer approval for SRS	Approve SRS		Project.SRS	-
	Customer	Approve SRS	Place approved SRS under configuration management		Project.SRS	-
	Team Leader	Place approved SRS under configuration management	Baseline approved SRS	SCM	Project.SRS	Submit artifact
	Team Leader	Baseline approved SRS		RM	Project.SRS	Update artifact

Figure 1 : Process Information in Tabular Format

3.4 Example

To better explain how the case-study was performed a sample that explains the analysis of a process definition is given below. “Obtain Commitment to Software Requirements” is a sub-process of “Requirements Engineering Process”. The process is defined as follows:

“Team Leader generates “Software Requirements Specification Document” using suitable “SRS Template” from Requirements Management tool. If needed, the generated UML diagrams and draft GUIs may be given as supplementary document to the SRS. PM achieves the customer approval for SRS. Once the customer signs off the SRS document, Team Leader places approved “Software Requirements Specification Document” under configuration and change control in accordance with the Configuration Management Plan. Team Leader baselines the approved requirements in Requirements Management Tool, in accordance with the Requirements Management Tool Guideline.”

This process is analyzed and components of it are identified to produce the table depicted in Figure 1. A model using the BPMN notation has also been developed to visualize the process. This model is available in Figure 2. With these representations of the process, we have reached the following conclusions:

- The process involves multiple interactions with CASE tools.
- The flow of the process does not include decisions. It does not require decision-making and creativity. It is a structured process.

- The interactions between users and CASE tools are simple. They consist of the following simple actions: get, submit and update artifact. The interactions are focused on a single artifact, the SRS document.

Because of the multiple, simple CASE tool interactions and the existence of a rigid structure in the definition; this process is labeled as suitable for automation.

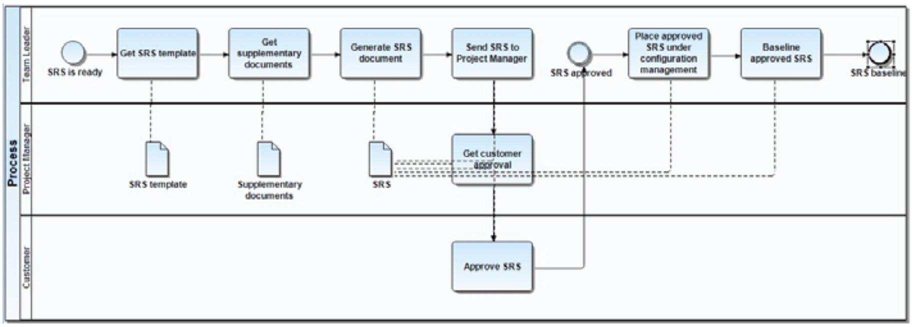


Figure 2 : BPMN Model

4 Results

For this case study, we have observed process definitions in 5 process areas, related guidelines and referenced documents. We can summarize our findings as follows:

A total of 110 process definitions were investigated. 88 of these 110 process definitions contain interactions with CASE tools. The distribution of process definitions with CASE tool interactions with respect to process areas is given in Table 1. 54 of 110 process definitions (49%) we have analyzed are labeled to be completely creative (unstructured) processes, including review, approval, analysis, design and development activities. This is an acceptable percentage considering the nature of the software organizations. These processes are thus considered not suitable for our ultimate goal of process automation. 30 of these 110 processes (27%) have multiple CASE tool interactions and structured (non-creative) simple user actions sequences. These are interpreted (subjectively) as presenting partial automation opportunities, through interactions with CASE tools. The distribution of process definitions that provide automation opportunities is given in Table 2. Counting only those processes including CASE tool interactions, the ratio raises to 34% (30 out of 88).

Table 1. CASE tool use

Process Area	# of proc. def.	With CASE tool use
PM	25	11
CM	17	11
RE	11	11
TS	13	13
VV	18	17
Docs/Guideline	26	25
Total	110	88 (80%)

Although PM process area was included in our analysis, it does not contain any processes with multiple CASE tool interactions and structured (non-creative) actions. If we remove the PM processes, the percentage increases to 35.3% (30 out of 85 processes).

Table 2. Distribution of processes providing automation opportunities

Process Area	# of process definitions	Suitable for automation
PM	25	0
CM	17	5
RE	11	4
TS	13	2
VV	18	1
Docs/Guidelines	26	18 (60%)
Total	110	30 (27.2%)
Total (CASE tools)	88	30 (34%)
Total (without PM)	85	30 (35.3%)

5 Discussion

These results are quite significant since almost one third of all process definitions provide automation opportunities. It should be noted that these processes are designed without considering any of these possibilities. Exploitation of these would increase the numbers in significant ways. Almost 60% of all automation opportunities presented are defined in guidelines and referenced documents. These documents outline general usage of CASE tools. Their content is developed considering the facilities provided by such tools.

Although on the process level the numbers are significant, we observed that the amount of structured actions in these processes is relatively low. We believe the reason is the lack of consideration given to facilities and integration potential provided by CASE tools, coupled with possible focus on provision of general guidelines to creative work. This may have resulted in the exclusion of structured work deliberately from process definitions. Of these 30 processes, the execution frequency differ greatly from “once per change request” to “once per project”. This corresponds to frequency values ranging from several times a day to once per year. The distribution of process execution frequency is given in Table 3. The value for “per project” execution (60%) is quite significant. We observed that processes executed once per project like project initiation, project closure, and intermediate project milestones have well structured definitions that provide automation opportunities.

Table 3. Process execution frequency

Frequency	# of processes definitions
Per project	18 (60%)
Per SRS release	5
Per SDD release	1
Per release	3
Per build	1
Per CR	1
Per requirement	1
Total	30

All process definitions we have analyzed contain simple interventions by user like adding an artifact to a repository or changing attribute values. Not surprisingly, these simple interventions did not have actions that need formalizations as data-mappings. This lack of complex actions can be attributed to the lack of consideration given to possible automation opportunities.

In this case study, we have found out that CASE tools are used extensively in the organization. It is directly observable from the process definitions that almost 80% of the processes include interactions with CASE tools. These tools however are not very well integrated to each other so they act as isolated process automation centers. In addition to this, we have seen that software development process definitions include structured activities that can be represented in process models, and can constitute a basis for process automation.

Through the case-study we have understood that, most of the work performed in a software development organization is non-structured and have creative elements. This could happen to be the reason why software process automation efforts have not been very successful. However, those processes also contain structured actions, and coupled with CASE tool interactions, provide automation opportunities. Structured actions we have observed in the process definitions differ greatly from those which are executed once per project initiation to those which are executed multiple times a day.

6 Conclusions

Based on the data from our case-study, we have gathered evidence supporting our vision that parts of software development processes can be automated through integration of CASE tools which individually provide isolated islands of automation. Based on the same results, we also believe that process definitions will be quite different if rigorous consideration to automation opportunities is given.

As a future work, we plan to develop a process model based methodology to exploit CASE tools and their integrations as a process automation infrastructure.

7 Literature

1. Curtis, B., Kellner, M. I., and Over, J. 1992. Process modeling. Commun. ACM 35, 9 (Sep. 1992), 75-90.

2. Tsalgatidou, A. and Junginger, S. 1995. Modelling in the re-engineering process. SIGOIS Bull. 16, 1 (Aug. 1995), 17-24.

3. Osterweil, L. 1987. Software processes are software too. In Proceedings of the 9th international Conference on Software Engineering (Monterey, California, United States). International Conference on Software Engineering. IEEE Computer Society Press, Los Alamitos, CA, 2-13.

4. Brown, Alan W., Feiler, Peter H., and Wallnau, Kurt C. Understanding Integration in a Software Development Environment. Tech. Rept. CMU/SEI-91-TR-31, ADA248119, Software Engineering Institute, Carnegie Mellon University, December 1991.

5. A. Christie, "A Practical Guide to the Technology and Adoption of Software Process Automation", Software Engineering Institute, CMU/SEI-94-TR-007, March, 1994

6. E. Kapsammer, T. Reiter, W. Schwinger, Model-Based Tool Integration - State of the Art and Future Perspectives, Proceedings of the 3rd Int. Conference on Cybernetics and Information Technologies, Systems and Applications (CITSA 2006), Orlando, USA, 20-23 July 2006.

7. F Altheide, S Dörfel, H Dörr, and J Kanzleiter. An architecture for a sustainable tool integration framework. In the Workshop on Tool Integration in System Development (TIS 2003), 2003. the 9th European Software Engineering Conference and 11th ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE 2003).

8. P. A. Oberndorf, The Common Ada Programming Support Environment (APSE) Interface Set (CAIS), IEEE Transactions on Software Engineering, v.14 n.6, p.742-748, June 1988.

9. M. J. Anderson, B. D. Bird: An evaluation of PCTE as a portable tool platform, Proc. Of the Software Engineering Environments Conference, July 1993.

10. Rony G. Flatscher, Metamodeling in EIA/CDIF---meta-metamodel and metamodels, ACM Transactions on Modeling and Computer Simulation (TOMACS), v.12 n.4, p.322-342, October 2002

11. OMG's CORBA Website, http://www.omg.org/corba , Last checked on: March 21, 2009

12. Object Management Group, Open Tool Integration Framework Request for Proposal, http://www.omg.org/docs/mic/04-08-01.pdf , Last checked on: March 21, 2009.

13. F Corradini, L Mariani, and E Merelli. An agent-based layered middleware as tool integration. In the Workshop on Tool Integration in System Development (TIS 2003), 2003. the 9th European Software Engineering Conference and 11th ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE 2003).

14. A. Earl: Principles of a Reference Model for Computer Aided Software Engineering Environments, Proc. Of the Int. Workshop on Software engineering environments, Springer-Verlag, New York, USA, 1989.

15. J. A. Bergstra, P. Klint: The Discrete Time ToolBus – a software coordination architecture. Coordination Models and Language, LNCS# 1061, 1996

16. K M Hansen. Activity-centred tool integration. In the Workshop on Tool Integration in System Development (TIS 2003), 2003. the 9th European Software Engineering Conference and 11th ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE 2003).

17. A. Schürr, H. Dörr: Introduction to the special SoSym section on model-based tool integration, Journal on Software and Systems Modeling (SoSym), Springer-Verlag, 4(2), May 2005.

18. Kaiser, G.E., Barghouti, N.S. And Sokolsky, M.H. Preliminary experience with process modeling in the Marvel software development environment kernel. In Proceedings of the 23rd Annual Hawaii International Conference on System Sciences, Vol. II – Software Track. IEEE Computer Society, Wahington, DC, 1990, pp. 131-140.

19. Kaiser, G. E. and Feiler, P. H. 1987. An architecture for intelligent assistance in software development. In Proceedings of the 9th international Conference on Software Engineering (Monterey, California, United States). International Conference on Software Engineering. IEEE Computer Society Press, Los Alamitos, CA, 180-188.

20. Peiwei Mi , Walt Scacchi, Process Integration in CASE Environments, IEEE Software, v.9 n.2, p.45-53, March 1992

21. Eclipse Application Lifecycle Framework (ALF) Project, http://www.eclipse.org/alf/ , Last checked on: March 21, 2009.

22. ALF Architecture, http://www.eclipse.org/alf/alf_architecture.php , Last checked on: March 21, 2009.

23. Sheth, A., Georgakopoulos, D., Joosten, S., Rusinkiewicz, M., Scacchi, W., Wileden, J., and Wolf, A. (1996). Report from the NSF Workshop on Workflow and Process Automation in Information Systems, Technical Report No. UGA-CS-TR-96-003, University of Georgia.

24. Kenneth R. Abbott , Sunil K. Sarin, Experiences with workflow management: issues for the next generation, Proceedings of the 1994 ACM conference on Computer supported cooperative work, p.113-120, October 22-26, 1994, Chapel Hill, North Carolina, United States

25. Moran, T. P., Cozzi, A., and Farrell, S. P. 2005. Unified activity management: supporting people in e-business. Commun. ACM 48, 12 (Dec. 2005), 67-70.

26. T. P. Moran, “Unified Activity Management: Explicitly Representing Activity in Work-Support Systems.” Proceedings of the European Conference on Computer-Supported Cooperative Work (ECSCW 2005), Workshop on Activity: From Theoretical to a Computational Construct (2005), http://www.daimi.au.dk/~bardram/ ecscw2005/papers/moran.pdf.

27. Blumenthal, R. and Nutt, G. J. 1995. Supporting unstructured workflow activities in the Bramble ICN system. In Proceedings of Conference on Organizational Computing Systems (Milpitas, California, United States, August 13 - 16, 1995). N. Comstock and C. Ellis, Eds. COCS '95. ACM, New York, NY, 130-137.

28. Moody, P., Gruen, D., Muller, M. J., Tang, J., and Moran, T. P. 2006. Business activity patterns: a new model for collaborative business applications. IBM Syst. J. 45, 4 (Oct. 2006), 683-694.

29. CMMI for Development, Version 1.2, http://www.sei.cmu.edu/publications/documents/06.reports/06tr008.html , Last checked on: March 21, 2009.

8 Author CVs

K. Alpay Ertürkmen

K. Alpay Ertürkmen is working toward the Ph.D degree from the Informatics Institute, Middle East Technical University (METU), Ankara, Turkey.
He is working as a Senior Technical Consultant. His research interests include software development process automation, CASE tools, CASE tool integration and process modeling.

Onur Demirörs

Onur's research focuses on software process improvement, software project management, software measurement, software engineering education, software engineering standards, and organizational change management. He managed research projects on developing software process improvement and modeling techniques for SMEs and for establishing and implementing business process modeling approaches for large scale software intensive system specification/acquisition. His work on software measurement particularly focuses on development of new functional size measurement methods and effort estimation techniques as well as establishing benchmarking datasets and validating current methods with field experiments. He worked as a consultant for a number of software developing companies to improve their processes based on ISO 9001, ISO 15504 and CMM and CMMi. He studies business process management for software organizations from measurement and modeling perspectives as well as process mining techniques. Currently he is the chair of Software Management Program at METU and Strategy Director at Bilgi Grubu Ltd.

Experiences of teaching code reviews for IT students

Zádor Dániel Kelemen[1,2], Katalin Balla[1,2]
1. Budapest University of Technology and Economics, Department of
Control Engineering and Information Technology,
Magyar tudósok körútja 2, 1117 Budapest, Hungary
{kelemend, balla}@iit.bme.hu
http://www.iit.bme.hu
2. SQI Hungarian Software Quality Consulting Institute,
József krt. 53, 1083. Budapest, Hungary
{kelemen.daniel, balla.katalin}@sqi.hu,

Abstract

This paper presents a case study on peer reviews carried out at Budapest University of Technology Software Testing Laboratory. Multiple literature sources were analyzed (models, standards, approaches) in order to understand the code review process and to develop an appropriate laboratory teaching material for both beginners and experienced students. The teaching material was tested first time, and feedback was collected in three different ways. The paper presents the structure of a laboratory material, the way of performing it, the resulted data analyzed and concludes with lessons learned and future improvement suggestions.

Keywords

code review, peer review, IEEE 1028, PSP, pair programming, multi-model SPI

1 Introduction

Software quality and software testing became more and more emphasized in Hungary at both multinational and Hungarian companies. This generates the concrete need for continuous improvement of teaching software quality and testing techniques. In order to satisfy this need, the Department of Control Engineering and Information Technology of Budapest University of Technology and Economics tries to improve its courses developed in 2001-2003 connected to the topics: “Software Testing” and “Software Quality and Management” , semester by semester. These courses were developed for students in informatics in their 4th and 5th year of study, right now being used in the new MsC courses for Control Engineering and Information Technology.

While the software quality and management course is more managerial and focuses on software processes, the course on software testing is more technical and product-oriented. Due to the different orientation of the courses we decided to include different approaches of peer reviews into our courses. A software requirement specification peer review material has been developed for software quality and management laboratory and a code review for software testing laboratory.

In this paper we present some elements of code and peer review, focusing on the lessons learned during using these methods in the software testing laboratory. Our article is about peer reviewing on course materials, as well.

2 Why reviews?

Besides the fact that peer reviews are present in the software industry, they are more widely known in other fields of industry, and it has been a touchstone of scientific methods since the 20th century. In software industry peer reviews had a wider acceptance since 1976, when Fagan wrote his famous article on design and code inspections [5].

Many *formal* (white-box and black-box) *testing techniques are already included* into our courses and laboratories, such as boundary-value analysis, equivalence-class testing, decision-table based testing, structural testing and others. However, researchers have shown that inspections and reviews also have a very significant effect on companies ROI [1][3] and a high percentage of saving in specification, design, code and test planning phases can be achieved [2] using them. Fagan shows that inspections detected 82% of the faults. In measurements of D. F. Rico, inspections had the *second highest ROI%* (3,272%) in comparison with SW-CMM (871%), CMMI (173%), ISO 9001 (229%), TSP (2,826%) and PSP (4,133%) [1][15]. Further studies [20] and software testing books [21][22] also show that a significant increase in productivity and product quality can be achieved using reviews.

Various benefits can be achieved from different *stakeholder* views. Karl E. Wiegers describes such benefits of peer reviews done by developers, project managers, maintainers, quality assurance managers, requirements analysts and test engineers point of view [4].

Many standards and models consider peer reviews being important and focus on describing requirements and best practices for peer reviewing (such as SW-CMM [6], CMMI [7], ISO 12207 [8], IEEE 1028:2008 [9] or PSP [10]).

The International Software Testing Qualification Board (ISTQB) [23] considers inspections / peer reviews a basic static testing technique and it is required to know for the Certified Tester - Foundation Level exam [21][22].

Due to its high ROI, benefits for different stakeholders, and its prominent presence in the well-known and widely used quality standards and models, we consider that peer reviews must be present in university courses for future software developers and testers.

3 Methodology

The purpose of this mini-case study was to examine the problems encountered and benefits derived during a pilot lecture at software testing laboratory when code review process has been used in practice. Given the objectives, the limited time and human resources available, it was determined that a combined, mainly qualitative approach with some quantitative aspects should be used. The case study approach was chosen from the many qualitative approaches available [11][12]. During the case study, feedback was collected using informal conversations with / among students, and lecturer's observation. As an additional technique, a web-based questionnaire was used to gain easily collectable and comparable feedback from participants. We used this technique because of the limited time available for conversations (2 hours laboratory with 16 participants). Also because of the small number of participants for a survey, the questionnaire was considered only as an addition to our mainly qualitative approach. The questionnaire and conversations focused first on the student's opinion connected to the practical experiences gained during the laboratory. Data collected through the questionnaire was validated by comparison with those collected during informal conversations and observations.

4 Case study

Budapest University of Technology and Economics is a leading university in Hungary in the IT field, hundreds of IT students are graduating year by year. Software testing course and laboratory was an optional course until 2009 spring, when it was transformed to a main course of system engineering specialization. In the semester 2009 spring, 18 students are attending it. (From the 18 students 16 were present at the code review laboratory.) The laboratories were firstly developed by Katalin Balla and refined later by K. Balla, Z. D. Kelemen and G. Bóka.

It is usual, that some students of software testing laboratory are already working at Hungarian or multinational companies, in parallel with their university studies. They are usually software developers, software testers, project managers or other technical staff at their companies. Some of them have more years of experience in software industry, others are beginners. A significant number of students are involved in the R&D projects run at the University or in joint projects between the University and software companies. Therefore when developing courses and laboratory material, we have to consider participants from the both sides: needs of beginners and experienced developers must be taken into account as well. It is mandatory that each student got the basic programming skills during the first years of their study, but afterwards they evolved at different levels in different technologies. Their programming language skills are mostly C++, Java, C# or web and PHP oriented.

4.1 Designing the peer review

Taking into account the diversity of our students, the code review material was developed using literature review, brainstorming and expert consultation.

Having the CMMI [6] as a basis we use in teaching concepts of SPI, we still faced that it will be difficult to implement the peer review process looking only to CMMI requirements. CMMI defines very high level requirements for peer reviewing: preparation, execution and analysis, but a concrete methodology or a step-by-step process is not included in the model. Therefore a multi-model approach [16] was chosen: we tried to search further literature to find more specific and more practical descriptions of the peer review process. We have found dozens of material connected to peer and code reviews. We selected Fagan's inspections, PSP, materials of Karl E. Wiegers and IEEE 1028:2008. The selection was done analyzing the strengths / weaknesses of each approach. The main strength of PSP and Karl E. Wieger peer review descriptions was considered to be the concrete template they gave for code reviews, the strength of Fagan inspections was the basis it gives for reviews (eg. how to structure the reviews, what is the process of review) and the main strength of IEEE 1028:2008 was considered to be the concrete definition and comparison of different types of peer reviews (management reviews, technical reviews, inspection, walk-through and audit). The standard IEEE 1028:2008 compares these five types of reviews based on the following characteristics: objective, decision making, change verification, recommended group size, group attendance, volume of material, presenter, data collection, output, formal facilitator training, defined participant roles, use of anomaly checklist, management participants, participation of customer or users.

Taking into account the CMMI's point of view and having the focus on code reviews we found "technical reviews" and "inspections" the most appropriate for our purpose in terms of IEEE 1028:2008.

4.1.1 Designing the peer review checklist

Having the template of Karl E. Wiegers [17] and PSP [10] and looking to the most common mistakes of C++, Java and PHP programmers, we decided to develop two types of checklists: a general one and a programming language-specific checklist for code reviews. Based on the templates mentioned, we included general questions having the focus on structure, documentation, variables, arithmetic operations, loops, branches and others, in totally having 30 questions. Due to the time constraint, the programming-language specific questions were planned to be developed for the next laboratory.

4.1.2 Choosing a code to be reviewed

A continuous, one-week long brainstorming went among K. Balla, Z. D. Kelemen and G. Bóka regarding the problem to be reviewed. Ideas came about reviewing specific code parts, which include the most common programming errors, also about writing a small piece of code and then finding the errors in the peer's code using the checklist developed. After having experienced teachers included in the brain-storming, it was discovered, that a part of students struggle with creating long and difficult pieces of code during programming laboratories, therefore we decided to choose a very easy and small program in order to involve the beginners and experienced developers at the same level.

Looking at the possible solutions we decided to use a well-known example in software testing: the triangle problem, which is a very useful mini-program for teaching white-box and black-box testing techniques (see [13] for the specification of triangle problem). Its specification and code are small enough to be understood quickly and it is relatively easy to make a quick review on it.

4.1.3 Coordinating the way of work during the laboratory

The main constraint to the laboratory work was the time. This laboratory should be fit into a normal laboratory time which is 2x45 minutes + 2x15 minutes breaks. Other resources as PC-s, development environment with code highlighting was provided to students.

At the beginning of the laboratory, a 10 minute long presentation on the tasks to be done and code reviews was given to the students. Review roles and typical techniques were presented in brief, as well as a web-based review software [14], which was available to use during the laboratory.

The following starting material was provided to the students:

- task list for the laboratory (see appendix 8.1)
- specification of the triangle program (the specification can be found in [13])
- code review checklist (the questions were tailored from [17])
- code review summary report template (based on the template at [19])
- web questionnaire for getting student's feedback (can be found at [18]).

Then students were asked to form 2 persons groups and use the given development environment to develop the triangle problem in pairs in the first 20 minutes. Students had the opportunity to choose freely a programming language from the following: C++, Java and PHP. In the next 30 minutes they got to know the checklist, then they performed the peer reviews on the code of another group. Errors observed were noted in the checklist and commented in the code. Finally, they were asked to fill a review report using the template given, summarizing their observations in it. In order to get feedback from this pilot, informal conversations and a web questionnaire were employed.

5 Results

5.1 Observations and informal conversations

16 students were present at the laboratory, they formed eight pairs and all of them choose Java for coding the program. We observed that 20 minutes were enough to finish coding the triangle program with no major time deviations. After coding, pairs shifted their code to the pairs sitting on their right, this way creating a circle of reviews. In order to simplify the way of working, from the common review roles [4][5][7][9] only two were selected: authors and reviewers. The next 30 minutes were also enough to review the code, based on the predefined checklist with 30 predefined general (programming-language independent) questions. The laboratory has finished successfully, 5 minutes before the targeted timeframe (2x45 minutes). At the end of the laboratory, students sent their work via e-mail to the instructor which included their own code, the completed code review checklist, the code review summary report and the code they reviewed. They were also asked to complete the web questionnaire.

Based on the materials sent by the students, the following findings can be formulated: the minimum LOC was 58, the maximum was 107, while the average was 80. In total there were 6 serious (for which a second review process was needed) errors found by 3 pairs. The other 5 pairs have accepted the codes with minor suggestions. In addition to the serious errors, 10 minor issues were discovered, from which 6 were related to the program structure, 2 of them were documentation issues, one was connected to branches and one to checking program inputs.

Based on the informal conversations during the laboratory and breaks, students seemed satisfied with the timeline and the tasks of the laboratory, many students appreciated the availability of a checklist for doing such a review, others students considered the problem too simple.

5.2 Result of the questionnaire

The peer review questionnaire (see appendix 8.2) was divided into two parts: the first part was used to briefly assess the experience of students (7 questions), while the second part contained 9 questions related to the laboratory. From 16 students present at the laboratory 14 students completed the questionnaire.

The first part of the questionnaire showed that there were 5 students participating with no previous experience, the other 9 respondents have experience in software development between 1 and 8 years (for which the average was 1,74 years). 6 out of 14 respondents were working parallel with their studies, most of them at mid-size companies.

The second part of the questionnaire showed that 13 of 14 respondents considered the laboratory useful and everyone had enough time to perform both the coding and review tasks. Two rating (from 1 to 5) questions were used in this part. The first was about the contribution of pairs in improving the code quality, for which the average answer was 2,71, while the second one was related to the usability the of the checklist for which the average rating was 3,86.

In the questionnaire 3 students have mentioned that they would prefer also more specific questions in the checklist and 5 showed the need for a more complex code.

6 Conclusion

Based on the observations, informal conversations and results of the questionnaire, the code review laboratory proved to be something new and interesting to students for which they reacted in a very positive way; 14 of 16 students filled the optional questionnaire, and 13 of 14 considered the laboratory useful. The material sent back of eight pairs via e-mail has shown that both serious and simple errors can be discovered using even so simple and quick reviews.

There were students who commented that they liked the laboratory (the material and the way of working). This kind of comments appeared both via informal conversations and textually in the questionnaire. On the other hand, others complained about the easiness of the problem to be reviewed, and comments were made on the checklist used as well.

The average of rankings on the quality improvement of the code and usability of review checklist suggest us that two main improvements might be made:

1. changing the code to a more complex one

2. improve on the checklist by extending it with language specific parts.

A simple problem (and code) was included in order to avoid failures of understanding from the beginners' side. Even the beginners finished all their tasks in time, which suggest that a bit more complex problem could be included as well. Changing the code to a more complex one could affect the performance of beginners, but giving more time for the review excluding the coding phase could serve a good solution. However, using differentiation in code complexity (for beginners and advanced programmers) could be also an alternative solution.

The idea of including language specific questions into the checklist seems useful.. For doing so, most typical language-specific errors should be collected, included into the questionnaire and injected into the code.

A further issue connected to the code is the code size. In the students' opinion it also can be extended to have a length of 2-3 times the length of the actual one (which was 80 LOC in average).

As a summary: the expected combination of students was present at the laboratory, both experienced and beginners. Due to their diversity, it can be quite difficult to design a (code review) laboratory which is appropriate for everyone. However, a harmonization should be made, and we think the usage of a well-designed, continuously improved teaching material can satisfy both needs. In order to achieve this, we will continuously improve this training material based on the results of each iteration.

The method used by us (structured questionnaire, observation and informal conversation) in peer reviewing a course material, having both teachers and students participating in the review process will be useful in reviewing further course materials, as well.

7 Acknowledgement

We would like to thank Gábor Bóka for expert consultation and the class participants of Software Testing (spring 2009) for the discussions on peer reviews and for a large number of excellent suggestions.

8 Appendix

8.1 Laboratory Task List

Steps:

1. Choose a programming language (C++, Java, PHP) and form pairs, and look for a pair who had chosen the same programming language. If there is no such team, choose another programming language.

2. In the first 30 minutes every team should create the triangle program, based on the specification attached. (*02_triangle_prog_spec.doc*).

3. In the next 20 minutes review the work of another team using the code review checklist given (*03_code_review_summary_report.doc*).

4. Create code review summary report using the template given (*04_code_review_summary_report.doc*).

5. Please fill in the code review feedback form found at: http://zador.cmmi.hu/limesurvey/index.php?sid=48419.

8.2 Questions of the web feedback form

- Personal Questions
    1. Email address
    2. Year of study
    3. Practical Experience (years)
    4. Are you working simultaneously with your studies?
    5. Job type
    6. How much time (years) did you spent at your current job.
    7. Company size
- Peer Review Experiences
    1. Do you consider the code review useful (the code quality as positively affected)? If not why?
    2. How your team contributed to increase the code quality? (1-5)
    3. Have you had enough time to write the code? If not how much time would be enough?
    4. Have you had enough time to perform the review? If not how much time would be enough?
    5. How useful was the code review checklist? (1-5)
    6. How would you improve de code review checklist?
    7. Were the appropriate participants present at the review? If not what competencies are needed?
    8. How would you improve the review?
    9. Would you recommend anything else to improve the code review process?

Literature

[1] D. F. Rico: How to estimate ROI for inspections, PSP, TSP, SW-CMM, ISO 9000, and CMMI, The DoD Softwaretech News, vol 5, no 4, pp. 23--31. (2002)

[2] Harjumaa, L. Tervonen, I. Huttunen, A.: Peer reviews in real life - motivators and demotivators, Fifth International Conference on Quality Software, pp. 29--36 (2005)

[3] D.F. Rico: ROI of Software Process Improvement: Metrics for Project Managers and Software Engineers, J. Ross Publishing, Boca Raton, FL (2004).

[4] Karl E. Wiegers: Humanizing Reviews http://www.processimpact.com/articles/humanizing_reviews.html

[5] M.E.Fagan, "Design and Code Inspections to Reduce Errors in Program development", IBM Systems Journal, Vol. 15, No. 3 1976, pp.182-211

[6] Software Engineering Institute - Capability Maturity Model for Software (CMM) http://www.sei.cmu.edu/cmm/ (1993)

[7] Software Engineering Institute, – “CMMI for Development, Version 1.2”

http://www.sei.cmu.sedu/publications/documents/06.reports/06tr008.html (2006)

[8] ISO/IEC 12207-95 – “Information technology - Software life cycle process” (1995)

[9] IEEE 1028:2008 - IEEE Standard for Software Reviews and Audits (2008)

[10] Software Engineering Institute: Overview of Team Software Process and Personal Software Process. http://www.sei.cmu.edu/tsp/ (2009)

[11] M. Saunders, P. Lewis, A. Thornhill, Research Methods for Business Students, 2nd ed., Financial Times Prentice Hall, London, 2000.

[12] R. K. Yin: Case Study Research: Design and Methods, ISBN: 978-1-4129-6099-1 (2008)

[13] P. C. Jorgensen: Software Testing: A Craftman's Approach, CRC Press, ISBN 0849308097 (2002)

[14] Code Collaborator: http://smartbear.com/codecollab.php (2009)

[15] D. F. Rico: Practical Metrics and Models for Return on Investment (http://davidfrico.com/rico03a.doc)

[16] Kelemen Z. D., Balla K., Trienekens J., Kusters R.: Towards Supporting Simultaneous Use of Process Based Quality Approaches, in Conference Proceedings of ICCC 2008, paper id: 69 (2008)

[17] Karl E. Wiegers: General Checklist for Code Reviews Template

http://www.processimpact.com/reviews_book/code_review_checklist.doc

[18] Z. D. Kelemen: Code Review Web Questionnaire http://zador.cmmi.hu/limesurvey/index.php?sid=48419

[19] Karl E. Wiegers: Inspection Summary Report Template

http://www.processimpact.com/reviews_book/inspection_summary_report.doc

[20] T. Gilb, D. Graham: Software Inspection, Addission Wesley (1993)

[21] D. Graham, E. Veenendaal, I. Evans, R. Black: Foundations of Software Testing (ISTQB Certification) ISBN: 978-1844803552 (2007)

[22] B. Hambling, P. Morgan, A. Samaroo, G. Tomphson, P. Williams: Software Testing – An ISEB Foundation, ISBN: 978-1-902505-79-4 (2008)

[23] ISTQB homepage http://www.istqb.org

9 Author CVs

Zádor Dániel Kelemen

Zádor Dániel Kelemen is a software quality and SPI consultant at SQI – Hungarian Software Quality Consulting Institute Ltd. He has been working in assessment and process improvement projects since 2005.

He is a PhD student at Budapest University of Technology and Economics (BUTE), Hungary, Dept. of Control Engineering and Information Technology in cooperation with Eindhoven University of Technology (TU/e), The Netherlands, Dept of. Industrial Engineering & Innovation Sciences, Information Systems Group. His PhD research focuses on software quality, especially on using multiple quality approaches in a synergic way and modelling process-oriented quality approaches. He has been participating in teaching the courses “Software Quality Management” and “Software Testing” at BUTE since 2006.

After finishing his (BSc) computer technology studies in 2002 at "Gh. Asachi" Technical University of Iaşi, Romania, he obtained his MSc degree from Budapest University of Technology and Economics (BUTE), Hungary in 2006 in IT engineering. The topic of his diploma-work was: "Supporting the use of the CMMI model".

He is an experienced web developer, he participated in several web-related and SPI-related R&D projects as a developer, researcher or technical leader. He has publications in Hungarian and international conferences in the topics connected to software quality, software process improvement, verification and validation.

Katalin Balla

Katalin Balla graduated as an informatician in 1984 from "Babeş Bolyai" University of Science Cluj, Romania. She worked as a programmer and a software system engineer until 1990, when she left to Bucharest, becoming a scientific columnist at the Hungarian newspaper "A Hét" and at Radio Bucharest. She attained post-graduate studies in software engineering at Technical University Budapest in 1993, followed by Ph.D. studies between 1994-1997. She obtained her Ph.D. in 2001 at the Technical University of Eindhoven, the Netherlands, in the field of software quality.

Mrs. Balla worked as a quality director at IQSOFT / IQSYS Ltd., for 11 years (1993-2004). She conducted all the process improvement projects at the company (introducing a company-wide project management methodology, followed by development of ISO 9001:1994 and ISO 9001:2000- conform quality management systems, the company's CMM-based software process improvement project and informal assessment).

Mrs. Balla is a lecturer at Technical University Budapest since 2001, where she developed and teaches a course in software quality management. Beginning with September 2007 she is an associate professor in the Department of Control Engineering and Information Technology.

In March 2004 Katalin was a founding member of SQI-Hungarian Software Quality Consulting Institute Ltd., where she has the job of managing director. She continues working as a consultant in software process improvement. She is a qualified SCAMPI Lead Assessor, ISO 9001:2000, Bootstrap and SPICE auditor.

She participated in many international research projects. She has a large number of scientific publications, and a presence in Hungarian and international conferences.

Tool Support for Collaborative Software Process Authoring in Large Organizations

Jeanette Heidenberg (EB, Elektrobit Corporation)
Petter Holmström (Åbo Akademi University)
Ivan Porres (Åbo Akademi University)

Abstract

Contrary to traditional practices, we consider that software processes should emerge and evolve collaboratively within an organization. In this article we present our vision of collaborative process authoring, we evaluate and discuss how existing process authoring tools suit this vision and we suggest a number of improvements to these tools to facilitate the deployment of a collaborative process authoring method in a large and geographically distributed organization.

Keywords

Software process authoring, software process improvement

1 Introduction

As the corporate world is growing increasingly aware of knowledge as an invaluable asset, the software process development follows suit. In large organizations, the software processes may grow into large and complex descriptions of the method know-how of the organization. This holds especially true if the products developed are large and complex, such as telecommunications or automotive systems in our case. A geographically distributed development environment further exacerbates the complexity. Process descriptions may be created and maintained using standard productivity software, but in our experience, specialized software process authoring tools are needed for maintaining complex, dynamic software process descriptions.

Contrary to traditional practices, we consider that software processes should emerge and evolve collaboratively within an organization. The time of the process developer in the ivory tower is coming to an end. The software process of the future should be the result of a collaborative effort of the software professionals of the company, sharing their best practices, methods and learnings.

However, we have found challenges implementing this tenet in practice, especially due to the lack of proper tool support. In this article we present our vision of collaborative process authoring, we evaluate and discuss how existing process authoring tools suit this vision. Finally, we suggest a number of improvements to these tools that would facilitate the deployment of a collaborative software process authoring method in a large and geographically distributed organization.

2 Collaborative Process Authoring

The main purpose of a software process is to support the people involved in the development of software products. There may be other, secondary purposes as well, such as proving the maturity of the company to a potential customer, or producing a paper trail for legal purposes. In the best case, these secondary purposes will automatically follow from a good process description. To the extent that they do place additional requirements on process descriptions, however, we address this issue when necessary.

In this section, we discuss our vision of good software process descriptions and the implication of our vision on the required features of process authoring tools.

We consider that a good software process description performs the following three tasks: communicate, remind and learn. The first task of a process description is simply to communicate the process to the project staff. Whenever a new member joins the team, you can point to the process description and say: “This is how we have agreed to work.” In order to serve this purpose, a process description will not consist of only work instructions, but also include guidance in the form of instructions, templates and examples.

The second task is to serve as a reminder for the project members, allowing them to check the process during the execution of the project. This differs from the first task in the way in which the process is presented. When serving as a reminder, the process description needs to display the specific information the reader needs now, rather than present the reader with the entire process.

The third task is different from the others in the direction of information flow. The process description should be able to learn from the experience of the projects. The process description can then evolve over time based on the experience and feedback provided by the projects using the process. As such, the process description can serve as a means for organizational learning as defined in [9].

In order for a software process description to fulfill its tasks of communicating, reminding and learning, it needs powerful tool support. Although it is possible to edit and maintain short process description documents using standard productivity software such as a word processor, this approach does not scale up to large processes. Process authoring tools such as Eclipse Process Framework Composer [3] and IRIS Process Author [4] simplify greatly the task of creating and maintaining large process descriptions.

In the following, we discuss the three tasks of a software process description separately. We argue for the necessity of each of them and provide a set of requirements we propose for process authoring tools pertaining to support software process descriptions in their task of communicating, reminding and learning.

2.1 Communicate: Disseminate Process to All Interested Parties

The software process of an organization encompasses the method know-how of that organization. This know-how needs to be efficiently disseminated to the project staff in order to be of use.

In our experience, processes are often considered to be an extra burden on top of the "real" work of developing software, so few software developers make an extra effort to observe the software process if it is not easily accessible and clearly communicated. For this reason, we often see traditional process descriptions, consisting of a large set of documents either in paper form or represented, e.g., as a set of PowerPoint slides, failing to be observed by the project staff.

Kellner [11] defines a set of basic requirements on process modeling tools, which we consider to cover the task of communication well. These include requirements on visualization, support for different viewpoints, multiple levels of abstraction, the use of a formal syntax, handling of multiple variants and versions as well as various analysis capabilities. As Kellner formulated his requirements already in 1988, most of them are already well addressed by existing tools. For this reason, we do not specifically list requirements for communicating software process descriptions, but rather focus on the more difficult tasks of learning and reminding. For further details, we refer the reader to Kellner [11].

2.2 Learn: Collaboration and Tailoring

Ideally, the process description embodies the best practices and lessons learned in the organization, in order for future projects to learn from the successes and mistakes of earlier ones [8]. The evolution of the process then becomes a collaborative effort of the company's method experts, i.e., the project staff. A process description can in that way be a dynamic knowledge base encompassing the process experience of the entire organization. As such a knowledge base can grow quite large, it is also important to have the possibility to tailor the process for the specific needs of each project. In this section, we discuss the needs of collaborative process development and process tailoring.

In order for a process description to be collaborative, it should be possible to comment on and annotate it in a distributed manner by anyone using the process. This is quite a large step away from previous paradigms, where a small group of process experts held the exclusive responsibility for updating the process description [14]. This requirement does not only have a significant impact on usability, but also on synchronization mechanisms in order to avoid inconsistencies arising from concurrent updates. Furthermore, there is often a need to make a distinction between parts of the process that are normative and officially released on the one hand and parts of the process that constitute guidance and are fully collaborative on the other hand. One example of such a need is if the software process is required to be appraised with regards to process maturity models or standards such as CMMI [1] or SPICE [5].

We summarize our collaboration issues in the following three requirements.

Learn 1 Comments and annotations possible for any user.

Learn 2 Support synchronized updates.

Learn 3 Differentiate between normative and collaborative versions of the process.

Every software project is unique. Over the years, many process models have been proposed as solutions for the software crisis and the ever increasing need for efficiency and productivity in software development projects. But no one process model has yet provided the perfect solution. In the end, there are no silver bullets that can make every project succeed. In our experience, the specific process needs of a project arise from parameters such as context, criticality, problem domain, size, business model and the experience and personality of the people involved. The agile [12] and lean [16] communities today emphasize the need for the process to be defined by the team itself. It would be inefficient to always start your process definition from scratch, though. For this reason, process engineering today often consists of tailoring the contents of an existing process repository to the needs of your specific setting [15]. In this way you combine the need for uniqueness with the power of reuse. The success of a process description then comes to depend on how easy it is to tailor.

In order for a process description to be tailorable there has to be a clear mechanism for extending, narrowing and redefining process content. When this is done, the need for version handling of both the original process description and the tailored process description becomes evident. We need to be able to state what version of the normative process the tailored one is based on. Furthermore, if you consider this feature in combination with the collaborative feature, it becomes clear that the connection from a tailored version of the process description to its original version needs to be maintained. If this is not the case, the collaborative work will be in the context of the process instance of the project in question only, and the lessons learned will be limited to the project in question, thus defeating the original purpose of organizational learning. Fig. 1 depicts these dependencies between the normative, the tailored and the collaborative versions of the process description.

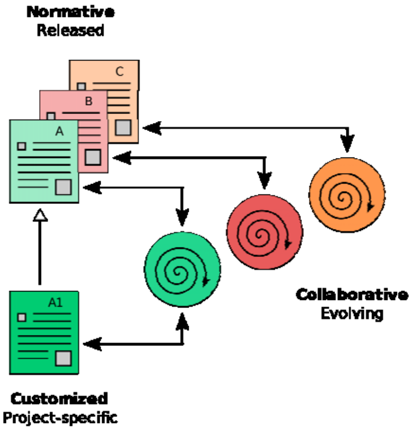


Fig. 1. The relations between normative, collaborative and customized process descriptions.

We summarize these issues with software process tailoring in the following requirements.

- **Learn 4** Support version hand ling of the process description.
- **Learn 5** Maintain traceability between original and tailored process descriptions.

2.3 Remind: Providing the Information the Reader Needs Now

In order for a software process description to serve as an efficient reminder, the content has to be adapted to the specific needs of the reader. A software tester will not want to read 40 pages on requirements engineering and software implementation before finding the parts relevant to testing. So the reader has to be given the possibility to access the process differently depending on his or her current role. On the other hand, the tester may be interested to know what other roles he or she should collaborate with on the activity currently under way or on the artifact under development. So the reader also has to be given the possibility to access the process differently depending on his or her current activity and artifact.

The possibility to access the software process description from different viewpoints was listed as a requirement already by Kellner [11] and has been further developed since. Today, it is a common feature in process authoring tools, due to the fact that many tools implement the standard meta-model for software and systems process engineering (SPEM) [13] as published by the Object Management Group. A meta-model describes the structure of the repository used by a process authoring tool to store process descriptions. We consider this to be a positive development that, however, does bring with it some drawbacks.

The standard meta-model is quite complex, which may lead to situations where the software process description itself gets unnecessarily complex. This in turn may necessitate the creation of company specific tool usage guidelines, which specify what parts of the meta-model should be used. These issues encumber the creation of process descriptions that may serve as efficient reminders, since it becomes difficult for the readers to find the information they need.

For this reason, we consider that process authoring tools should provide a way of easily customizing the meta-model to meet the needs of the users. The GUI itself may also need to be customized in order to better present the customized meta-model. Since usage guidelines may still be needed, it would be beneficial if companies could integrate their usage guidelines closely with the tool.

We summarize these aspects of a software process description serving as an efficient tool for reminding in the following requirements.

- **Remind 1** Enable access to the process through different views based on the need of the user.
- **Remind 2** Support customization of the meta-model.
- **Remind 3** Support integration of company specific usage guidelines in the tool.
- **Remind 4** Support customization of the GUI.

3 Issues with the Current Process Authoring Tools

With the requirements outlined in Section 2 in mind, we have evaluated some of the most popular open source and commercial process authoring tools available today, namely Eclipse Process Framework Composer1 (EPC) [3], Rational Method Composer2 (RMC) [6] and IRIS Process Author (IPA) [4]. In addition, we also included an in-house process authoring tool in the evaluation. The results are summarized in Table 1. We have found that that these tools have some inconveniences that prevent them from fully meeting all our requirements. In the following, we are going to discuss the problems encountered.

Table 1. Tool evaluation results (C=Compliant, PC=Partially compliant, NC=Not Compliant)

Req.	In-house	EPC	RMC	IPA
Learn 1	C	PC	PC	C
Learn 2	PC	PC	PC	C
Learn 3	C	NC	NC	C
Learn 4	C	C	C	C
Learn 5	NC	NC	NC	NC
Remind 1	C	C	C	C

[1] Version 1.2

[2] Version 7.2

Remind 2	C	NC	NC	PC
Remind 3	NC	NC	NC	NC
Remind 4	C	NC	NC	NC

3.1 Learn: Collaboration and Version Handling

In the first requirement related to the task of learning (Learn 1), we suggest that collaboration between process authors and process users demands that any user should be able to comment and annotate the process. We now proceed to present our findings in respect to this requirement.

Many tools are still designed around the traditional process engineering principle, where the processes are designed by a dedicated process engineering team and then exported to the software engineering teams for adoption.

Of the evaluated tools, IPA has the best support for collaboration through its Process Central and Wiki features. Close behind follows the in-house tool that generates links to the company wiki from every process entity. Thus, anyone with access to the company wiki can contribute to the process. However, the changes are not automatically propagated from the wiki to the process authoring tool — they have to be imported manually (as is the case with IPA as well). EPC/RMC can be configured to provide a single link for collecting user feedback. It is not possible to specify different feedback links for different entities within the process.

In Section 2.2 we listed four more requirements which support a process description’s task of learning. These are related to various aspects of version handling (Learn 4). When it comes to authoring and documenting process descriptions, a good version control system is important for two reasons: synchronization (Learn 2) and version tracing (Learn 5). Version handling also helps differentiate between normative and collaborative versions of the process (Learn 3). We now proceed to present our findings in respect to these requirements.

Synchronization becomes an issue in all situations where several actors compete for a limited number of resources. In this case, several users must be able to work on the process simultaneously without having to worry about overwriting each other’s changes. This issue is normally resolved by using a locking mechanism. Two popular locking mechanisms are “Lock-Modify-Unlock” and “Copy-Modify-Merge” (examples of both mechanisms are presented in Chap. 1 in [7]).

Lock-Modify-Unlock prevents other users from editing a locked entity until the lock is released. This is acceptable if the size of the lock area is small enough. However, if the lock spans large areas of the process description, it effectively blocks collaboration, as only one user can work on the process description at a time.

Copy-Modify-Merge detects conflicts instead of preventing them. When a conflict is detected, the slower author has to merge his or her version with the one currently in the repository before it can be committed. However, if a general revision control system such as Subversion, CVS or ClearCase is used, this becomes a problem as these systems work with ordinary files. The slower author either has to revert his or her changes, update the working copy and re-add the modifications manually, or exit the tool and merge the changes on a raw code level (e.g. if the tool stores its data in XML-files). In either case, it is a time consuming and error prone process.

IPA has tried to address the synchronization issue by using a built-in Lock-Modify-Unlock mechanism that works on the process library level. Thus, it is impossible for two users to work on the same process library simultaneously, causing a possible block to collaboration. This problem can be worked around by making sure the libraries are small enough, which is a good idea in any case as it makes reuse easier as well. IPA handles version tracing by dumping the process descriptions to XML and storing them in external revision control systems. The XML dumps can then be re-imported into IPA as necessary.

The other evaluated tools all rely on external revision control systems for both synchronization and version tracing. Thus, they all suffer from the problems mentioned previously in this section.

3.2 Remind: User Interface Issues

As we outline in Section 2.3, a process description's function as a reminder relies on how efficiently the user finds the specific information needed at that particular point in time (Remind 1). This in turn depends on the suitability of the user interface on the one hand (Remind 4) and the suitability of the underlying meta-model on the other hand (Remind 2). In situations when the user interface and meta-model are overly complex, the user will need easy access to usage guidelines (Remind 3). We now proceed to present our findings in respect to these requirements.

Most tools use a standardized meta-model such as SPEM [13]. This has the benefit or reducing vendor lock-in and increasing portability between different process authoring tools. However, it has also led to vendors including in their tools all possible concepts that might be needed to model a process, resulting in overly detailed process descriptions and complex user interfaces. Few users will actually need all the functionality, which forces companies to introduce usage guidelines. This in turn makes the tools harder to use and scare off casual users whose feedback and opinions would otherwise have been very valuable.

The meta-model in most tools is fixed, i.e. the users cannot define custom entities or attributes. Even in cases where the meta-model allows customization in principle, the user interface does not support this in practice. Consequently, the users are forced to model their processes using the predefined process elements.

Of the evaluated tools, only the in-house tool supports scaling of the user interface, thanks to its customizable meta-model. IPA supports user-defined attributes, making it possible to make some customizations to the meta-model.

4 A Plea for a New Collaborative Process Authoring Tool

In Section 3 we listed issues with many modern process authoring tools. In this section, we are going to present proposals and ideas to resolve these issues, thereby improving the state of the practice.

4.1 Learn: Integrate Authoring, Publishing and Peer-Review in a Web-Based System

We believe that the key to continuous process improvement is collaborative process review and editing. To simplify this task we propose to integrate process publishing and authoring tools in the same web-based system. A clear precedent for this is the wiki, which has lead to massive collaborative efforts such as the Wikipedia. A wiki allows a user to switch from a reader to a contributor role simply by clicking a button placed next to the contribution to be edited or expanded. There are usually few (if any) limitations on when or by whom the wiki contents can be modified. However, contributions are reviewed afterwards by an editor and they can be edited or even discarded if they are not considered suitable.

In the context of authoring software process descriptions, we propose to use one single web-based application tool to author, communicate, comment and review software processes within an organization. Although it is possible to use a dedicated desktop-based process authoring tool in conjunction with a standard wiki to allow free-form discussions in an organization, we believe that an integrated web-based solution would really foster a collaborative effort where each user of a process description also becomes a contributor.

One of the main challenges when improving a process description collaboratively is reviewing and merging multiple contributions from different sources. Although it is possible to use text-based difference, merge and version control tools designed for source code (such as diff, patch and git) or text documents (such as the revision facilities in Microsoft Word), this may easily lead to inconsistent process descriptions. The underlying problem is that a software process has a graph structure, while it is often represented as a text (string). Also, complex process descriptions are often split into different files or pages.

Based on this and in response to the issues presented in Section 3.1, we consider that web-based process authoring and reviewing systems should implement a model aware diff-and-merge component. This component would allow a user to compare the differences between the two graphs representing two versions of the process description the model level rather than two text strings. The component would also allow a user to merge the two versions using a graphical user interface.

4.2 Remind: Scalable User Interface

We propose that every process also comes bundled with an embedded user interface configuration for the tool. The configuration would contain information about the attributes and entities that are to be used for the process, as well as specific usage guidelines as an optional element. That is, instead of having a complex user interface and a separate usage guideline document that instructs the users what to fill in and what to leave blank, the tool would only display the fields that are to be filled in and provide optional corresponding instructions inside the tool, as the user navigates the user interface (see Fig. 2).

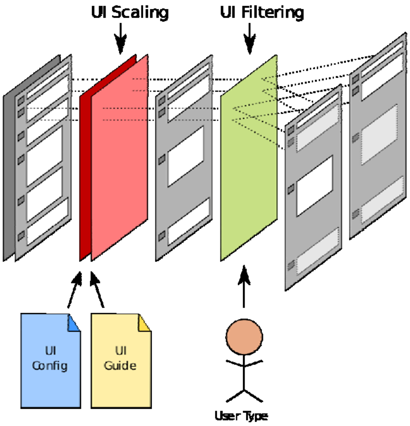


Fig 2. Scaling and filtering of the user interface.

In addition, we propose that the user interface is filterable depending on the needs of the user as described in Section 2.3 (see Fig. 2). A software engineer and a process engineer do not need access to the same information. This feature is already available in many of the tools3, but only in the exported process description. We propose that this feature is applied to the user interface as well.

The tool could come with a set of predefined user interface configurations for different types and sizes of projects. However, it should also be possible to customize these configurations in an easy way, e.g. similar to the content variability mechanisms of SPEM (see Sect. 6.3.4 in [13]).

[3] This feature is called *Views* in EPC/RMC.

5 Conclusions and Related Work

In this article we have discussed the importance of a collaborative effort in process authoring and what we consider are the main challenges in deploying this in practice and how these challenges could be overcome with proper tool support.

In order to meet the challenges of an industry that requires increased efficiency, agility and streamlining of the software development operations, we consider collaborative process authoring to be of vital importance.

We have presented three tasks for a software process: communicate, learn and remind. We consider that the existing authoring tools perform the communicate task well. However, when we consider the learn and remind tasks, they fall short. The reasons for this are that first, they have been designed with the traditional process engineering principle in mind, where the processes are designed by a dedicated process engineering team and then exported to the software engineering teams for adoption; and second, their vendors have included all possible process modeling concepts into them without proper filtering and scaling capabilities. We suggest that the tool vendors shift focus and concentrate on making their tools more collaborative, customizable and scalable to different process sizes.

Among the commercially available evaluated tools, IRIS Process Author (IPA) [4] stands out due to its innovative way of handling collaborative process development. Its Process Central and Wiki features come close to the needs we have outlined. The original assumption, however, still seems to be that process development is driven by a team of process engineers, whereas we propose a completely decentralized way of working where everyone is a potential process engineer. The linkage between the normative version and the Wiki version is limited in IPA to a top level link. We would prefer a link for each process entity. We have been informed, however, that this feature is included in the roadmap of IPA. It is still unclear how well this feature will interact with process tailoring, as outlined in Section 2.2. The feedback loop from the collaborative version to the normative one is also still unclear.

We can consider that the Cunningham & Cunningham wiki [2] has been used as a collaborative process authoring tool. Many software development approaches have been discussed and criticized in that web site, and this has been highly influential in the current thinking in agile software development. However, we consider that free wiki discussions do not replace a structured software process description, although attempts to remedy this situation have been suggested by Wongboonsin and Limpiyakorn [17]. Also, there are no distinctions between the normative version of the processes and the collaborative versions under discussion and review.

In their article “Enough Process - Let’s do Practices” [10], Jacobson et al. present similar concerns to those presented in this article. They suggest “a shift of focus from the definition of complete processes to the capture of reusable practices” and that “teams should be able to mix-and-match practices and ideas from many different sources to create effective ways of working”. In their article, they list some common problems that many software processes struggle with and provide some interesting ideas on how to solve them. We think that in addition to the ideas presented in this article, the ideas of Jacobson et al. should be taken into account when the next generation of process authoring tools is designed.

In conclusion, we believe that software process authoring in large organizations is moving in a collaborative direction. In order to support this positive development, we propose that the tool development effort needs to focus on collaborative authoring, on integrated process presentation and authoring as well as on more flexible tailoring of the process presentation.

6 Acknowledgements

This work was partly funded by the EB Flexi project and the Flexi EU-project of ITEA framework. The authors wish to thank Jari Partanen for his support and Martin Dusch and the members of the process authoring tool user group at EB for their input and support.

7 Literature

1. Carnegie Mellon Software Engineering Institute. http://www.sei.cmu.edu/cmmi/.

2. Cunningham & Cunningham Wiki. http://c2.com.

3. Eclipse Process Framework Web site. http://www.eclipse.org/epf.

4. IRIS Process Author Web site. http://www.osellus.com/IRIS-PA.

5. ISO/IEC 15504 (SPICE). http://www.isospice.com/categories/ISO%7B47%7DIEC-15504-Standard/.

6. Rational Method Composer Web site. http://www.ibm.com/software/awdtools/rmc/index.html.

7. B. Collins-Sussman, B.W. Fitzpatrick and C.M. Pilato. Version Control with Subversion. Published online, http://chestofbooks.com/computers/revision-control/subversion-svn/index.html, 2008.

8. Victor R. Basili and Gianluigi Caldiera. The experience factory: strategy and practice. Technical report, College Park, MD, USA, 1995.

9. Nancy M. Dixon. Organizational learning cycle : How we can learn collectively. Gower Publishing Limited, 1999.

10. I. Jacobson, Pan Wei Ng and I. Spence. Enough Process - Let's Do Practices. Journal of Object Technology, 6(6):41–66, July-August 2007.

11. M.I. Kellner. Representation formalisms for software process modelling. In Proceedings of the 4th international Software Process Workshop on Representing and Enacting the Software Process, pages 93–96, Devon, United Kingdom, 1988.

12. Robert C. Martin. Agile Software Development: Principles, Patterns and Practices. Prentice Hall, 2003.

13. OMG. Software & Systems Process Engineering Meta-Model Specification Version 2.0. Specification, Object Management Group, Inc., April 2008. OMG Document Number formal/2008-04-01.

14. P. Fowler, S. Rifkin. Software Engineering Process Group Guide. Technical report, SEI, September 1990. CMU/SEI-90-TR-024.

15. Oscar Pedreira, Mario Piattini, Miguel R. Luaces, and Nieves R. Brisaboa. A systematic review of software process tailoring. SIGSOFT Softw. Eng. Notes, 32(3):1–6, 2007.

16. Mary Poppendieck and Tom Poppendieck. Lean Software Development: An Agile Toolkit. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.

17. Jenjira Wongboonsin and Yachai Limpiyakorn. Wikipedia Customization for Organization's Process Asset Management. International Conference on Advanced Computer Theory and Engineering, pages 467–471, 2008.

8 Author CVs

Jeanette Heidenberg

Jeanette Heidenberg received her M.Sc. degree in Computer Science in 1999 at Åbo Akademi University. She has since worked as Software Designer and Architect for Ericsson (2000-2006) and Software Specialist for EB (2006-2009) in Finland. Since 2003, her main task has been process development, focusing on model driven and agile methods. She is currently pursuing her Ph.D. in Computer Science at Åbo Akademi University.

Petter Holmström

Petter Holmström is a master's student in Computer Engineering who will receive his degree in 2009. In his master's thesis, he investigated how well current software process authoring tools are able to conform to the requirements of a large organization. He currently works as a research assistant at Åbo Akademi University.

Ivan Porres

Ivan Porres received his M.Sc. degree in Computer Science in 1997 at the Polytechnic University of Valencia, Spain and in 2001 his Ph.D. in Computer Engineering at Åbo Akademi University in Turku, Finland. He is currently professor of Software Engineering at Åbo Akademi University where he performs his is research on software design languages and supporting tools, engineering software services and software process improvement.

An Enhanced Variability Mechanisms to Manage Software Process Lines

Tomás Martínez-Ruiz, Félix García, Mario Piattini
Alarcos Research Group, Information and Technology Systems Department
Escuela Superior de Informática, University of Castilla-La Mancha
Paseo de la Universidad, 4, 13071 Ciudad Real, Spain
{tomas.martinez, felix.garcia, mario.piattini}@uclm.es

Abstract

The characteristics of projects and organizations influence process development and must therefore be taken into account during process tailoring. Software process lines permit process adaptation to include these characteristics in processes. However, it is necessary to define suitable variability mechanisms for software processes, considering the elements that take part in them, their relationships and their evolution and improvement. An analysis of processes, the elements of which they are composed, and how they may vary, permit the variability mechanism used in product lines to be applied and enhanced to tailor software processes. Different types of variants and variation points can be distinguished according their use in tailoring processes. These can be used to obtain tailored processes by paying attention to the elements which will be varied, since the others are managed automatically, thus providing the means for them to be used through their improvement. The proposed mechanisms consider simplified tailored process creation whilst simultaneously ensuring transparent process consistence from the process engineer’s point of view.

Keywords

Software Process Lines, Software Process Tailoring, Variability Mechanisms, Variant, Variation Point, Dependences, Software Process Institutionalization

1 Introduction

Software development has evolved from artisan methods to become more engineering centred, resulting in the creation of software product lines [1]. This approach permits the management of families of products, which offers some advantages. On the one hand, software product lines make it possible to tailor products to the user's needs by making good use of both the similarities and differences between the products. On the other hand, the costs of creating products can be reduced, and the quality of the product line and its products can be improved while they are generated [1].

Bearing in mind that "software process are process too" [2], several software engineering techniques have been applied in software process, and this is termed as process engineering. Software process lines are thus created from product lines [3]. The use of this approach permits processes to be tailored to the characteristics of both projects (since "just as there are no two identical projects, there are no two identical processes in the world" [4]) and organizations [5], which is necessary for their survival [6]. Moreover, some process models, such as CMMI, specify software process improvement, which implies that processes need to be engrained in the way in which work is carried out in organizations [7]. In order to engrain, these processes must be tailored according the specific characteristics of each project and organization. The inclusion of organization characteristics in generic processes by means of tailoring is thus a key factor in engraining processes in organizations [8].

It is also necessary to learn about the adapted processes in order to improve the way in which the processes are tailored, and therefore improve the tailored process. Supporting the adaptation of processes by using well defined and well bounded variations implies that any changes are well-known and well controlled, which facilitates learning about their realization and how to improve it [8]. The mechanisms which permit variations are variants and variation points [9]. The former represent single parts that vary, and the latter are the parts of the process into which variants can be inserted. They also define dependences in order to carry out difficult tasks such as guaranteeing the consistence of the generated processes.

However, in order to ensure ease of consistence and to make its application possible by means of automatic tools, both a clear specification of the variability elements and a description of their behavior and the definition of the scenarios in which they may interact are necessary. Tailored processes can thus be simplified to select the variants which are most suitable for the organization and project, knowing that the process is always consistent.

In this paper we have analyzed the various dependences that may be present in a process line, and how these dependences affect such variants or variation points. This analysis has allowed us to classify both variants and variation points according to their behaviour and, what is more important, according to how the process engineer views them when s/he is about to tailor a process. Furthermore, both the view of process lines and the way in which consistence is ensured can be simplified.

Section 2 presents the state-of-the-art in variability in process and product lines and a summarization of SPEM. Section 3 summarizes the previous approach to manage variability. Section 4 presents the analysis carried out, which is illustrated through an example. Finally, our conclusions and future work are presented in Section 5.

2 State of the art

Software process lines are a new discipline and, to the best of our knowledge, very few works concerning variability in process lines therefore exist. In [9] a variability approach to process lines is presented, which will be summarized in Section 3. Furthermore, the Fraunhoufer IESE is working on variability in processes in order to customize software inspection, showing mechanisms based on variants over several process elements [10].

Several works concerning variability in workflows and business processes also exist. [11] describes how to use variability in software processes, since organizations must adapt their processes to survive

[6], in order to manage them as a process line [12]. Variability must be based on conceptual processes, in a similar manner to that of product lines [13]. According to [14] variability allows the propagation of best practices.

However, as process lines are based on product lines, it is possible to apply the software product line variability mechanisms proposed in literature to the modelling of variability in software process lines. A variation point is the place in which variability occurs [15], and it has the advantage of adding new variability implementations, by adding new variants [16], the concrete elements that are placed at the variation points, each of which implements this variability in a different way [17]. Both of them can be distinguish by using stereotypes [18]. According to [19], it is possible to specify dependencies and constraints between these elements. In [20] three abstraction levels for modelling variability with dependencies are presented, and [21] presents patterns with which to build, manage and manipulate variation points.

If process lines are to be modelled in a usable way, they must be based on process models. SPEM (Software Process Engineering Metamodel) [22] is the Object Management Group (OMG) proposal through which to create software processes by using methods. This proposal is one of those most used by industry because it is independent from any process model.

As is explained in [9], the new 2.0 version of the standard permits variability when methods are being modelled, or when method components are used to compose process activities. However, it does not, in itself, permit variability in processes. Furthermore, the mechanisms included in SPEM are not able to define the core process of the software process line, and the common elements of all the processes in the line cannot, therefore, be guaranteed. Finally, SPEM does not include specific variability notation to facilitate the use of variability mechanisms when processes are to be tailored. An extension of SPEM is thus proposed below.

3 Variability in Software Process Lines

An extension of SPEM which permits processes to vary is proposed in [9]. All the elements of this proposal can be included in a new package in the SPEM *ProcessLineComponents* metamodel(Figure 1). This new package uses elements from the SPEM *ProcessWithMethods* and *ProcessStructure* sub-packages, with the aim of adding the variation capability to the process constructors in both packages.

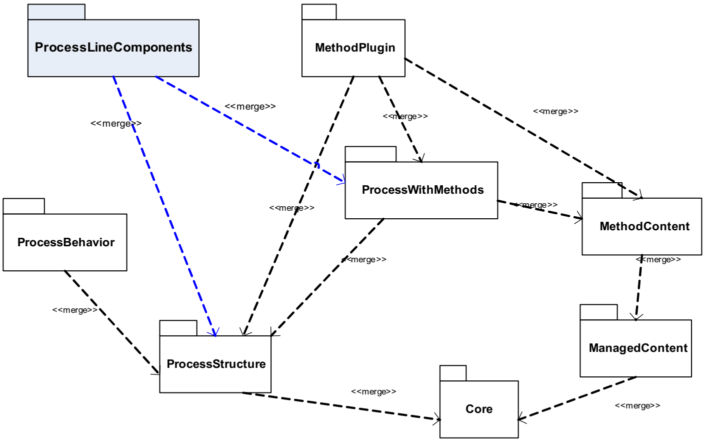


Figure 1: *ProcessLineComponents* sub-package in the SPEM metamodel

As is shown in [9], variability is basically managed by five elements. The first is the core process, which contains the common elements of all the processes of the process line with several empty points, the variation points, in which the variation occurs. The variants are the concrete elements which are inserted into the variation points and tailor the resulting process. The *LProcElement* abstract class is a super-class of both elements, as Figure 2 shows. Variants and variation points are also abstract elements, and must be concretized from different SPEM concrete process elements as Figure 2 shows, by using the SPEM activity.

The other two elements are dependences between variants and variation points, and the occupation relationship, used to represent which variant is occupying which variation point. Occupation is a key element because it makes it possible to convert variation points into process elements, giving support to variability.

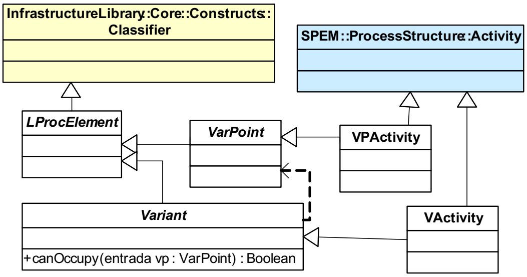

InfrastructureLibrary::Core::Constructs::
Classifier
SPEM::ProcessStructure::Activity
LProcElement
VarPoint
VPActivity
Variant
+canOccupy(entrada vp : VarPoint) : Boolean
VActivity

Figure 2: Metamodel of Variants and *VariationPoints* in the *ProcessLineComponents* package

Moreover, process elements such as activities, tasks, roles, and work products do not appear separately in the process models. They are related to each other. These relationships are therefore between variants and variation points as dependences. For example, a product can be an output of a task (the product is produced by the task), so a relationship exists between the task (source) and the product (target). When both the product and the task are variants, the relationships become dependences between variants. Dependences help us to ensure the consistence of the processes created from the process line, and they signify that if an element depends on the inclusion (or exclusion) of another one, then this dependence must be satisfied by means of including (or excluding) another element. It is important to know that dependences are defined in the opposite direction to element relationships because, in order to consider the target, the source must also be considered. Both elements are metamodeled in

Figure 3.

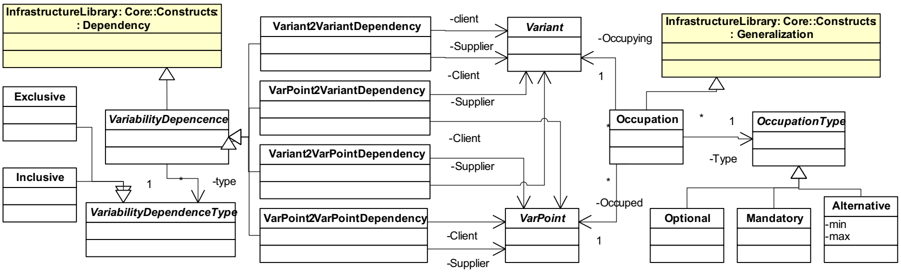


Figure 3: Metamodel of *Variability* Dependences and *Occupation* Relationship

3.1 Process Line Example

The aforementioned elements can be used to define a software process line with which to tailor processes [9]. We have defined a process line of the *Assessment and Improvement Software Processes* process. This process includes two activities, Assessment and Improvement, and we have defined variability in both of these. Both have a *VPWorkProduct* input to introduce variants of work products (*VWorkProduct*), and the Assessment activity includes a sucessor *VPActivity* (variation point of Activity) in which we can place variants of Activities (*VActivity*) (Figure 4-left). These points can be used to introduce new elements to tailor the process we are going to create. Figure 4-right shows that there are several variants related to the core process in order to occupy the variation points.

To create a tailored process, the variation points of the core process must be occupied by the variants from Figure 4-right. As regards the dependences between the variants (variant2variantDepences) Figure 4-right shows that this means that the Used Improvement Model will be produced by the Model Decision task which uses the Result presentation input work product, and the Improvement Manager role. If we wish to consider the Used Improvement Model it is, therefore, also necessary to considered the Model Decision, the Result Presentation and the Improvement Manager. The resulting tailored process is shown in Figure 5.

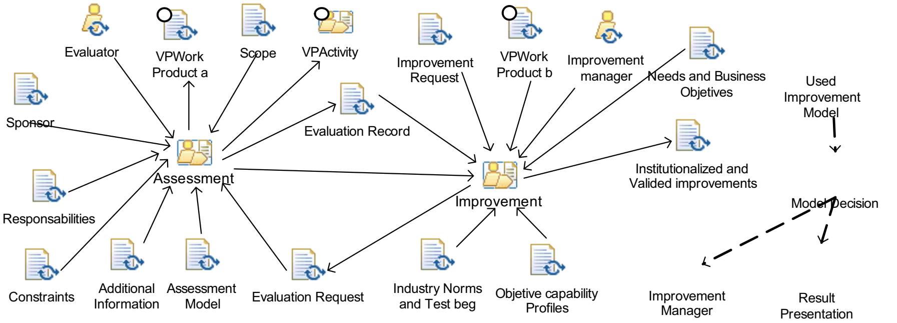


Figure 4: Assessment and Improvement core process (left side) and their variants (right side)

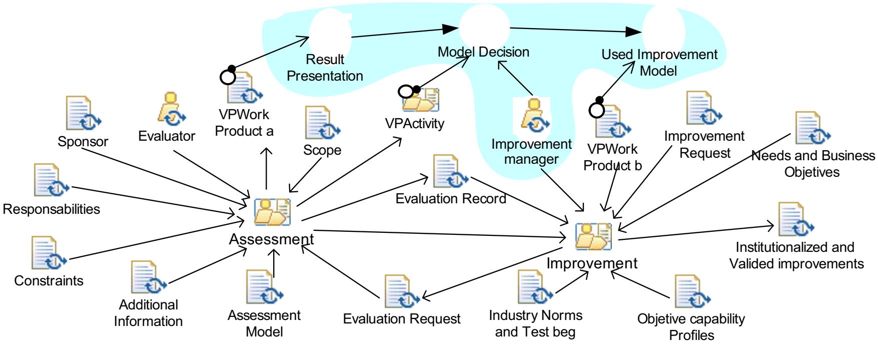


Figure 5: Process tailored by means of the occupation of its variation points with variants

Figure 5 shows a tailored process created from the process line of Figure 4-left, using the variants of Figure 4-right to carry out the “include the User Improvement Model into the Improvement Activity as input work product” variation. It is important to note that since the Improvement Manager role is already part of the core process, it is not included again. Furthermore, this variant cannot be included because the core process has no variation points of type role. This means that if the Model Decision VActivity depends on any other role variant which has not already been defined in the core process, neither the role variant nor the variants depending on it can be included.

4 Enhanced variability mechanisms to built consistent models

Previously we have stated dependences help to ensure the consistence of the tailor processes, and drive how to variants and variation points are related [23]. In the methodological framework we have development; the behaviour of these dependences are vary depending on the elements they related. The behaviours we have described are the following:

- *Variant-to-variant*: these must be used to guarantee that the variants which are going to be inserted do not include inconsistencies between them or the other variants previously inserted into the process when it is going to be tailored.
- *VariationPoint-to-VariationPoint*: these must be used to guarantee that the occupation of variation points are consistently related to the other variation points which are present in the core process and their occupation.
- *Variant-to-VariationPoint*: These allow variability propagation, i.e., when a variant occupies a variation point this may force other variants to be placed as a result of variant-to-variant dependences. However, it may sometimes be of interest to include variability in these other variants. For example, if we insert a task variant, it must be carried out by a role, but it may be of interest to define variability around the role that develops it. These dependences link variation points which are not included in the core process, but are grouped with variants.
- *VariationPoint-to-Variant*: These only signify that the default variant or variants occupy the variation point.

Furthermore, the first two dependences can be defined as either inclusive or exclusive. The first type implies that if the client element is considered, then the supplier element must also be considered. The second type implies that if the client element is considered then the supplier element must not be considered. However, only the last two dependences can be defined as inclusive. A new approach is presented by using a modification of the example from the previous section. The consideration of the aforementioned dependences enables us to carry out a more exhaustive description of the variants, variation points and the occupation relationship between them.

4.1 New Variants

Variants are components of process lines which permit variability. Considering that dependences affect related variants, we can distinguish between two types of variants:

- Root variants. These variants are not involved in any variability dependence as the supplier element, unless the client element of the relationship is a variation point. They may or may not be the client element of any variability dependency.
- Non-root variants: these variants are involved in at least one variability dependency as a supplier element.

The type of a variant is not an inherent characteristic, and may be changed. That is, a variant can be a root variant at one moment (since no variants depends on it), and another variant may later be defined which depends on the first. The first variant then becomes a non-root variant. Therefore, if a variant is deleted (because it will become obsolete) other variants can be converted from non-root variants into root variants.

Specific notations for each type of variant have been created in order to distinguish between them in graphic diagrams. Variants are represented by means of a “V” placed in the top left-hand corner of the SPEM element which is used to create the variant. Therefore, root variants use a black “V”, and non-root variants use a white “V”. Figure 6 shows this notation when applied to activity variants.

These icons and the dependences described in Section 4, are used to draw some related variants in Figure 7, which shows the difference between the root variant and the others (non-root).

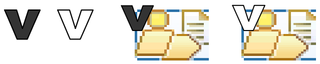


Figure 6: Icons to represent root and non-root variants

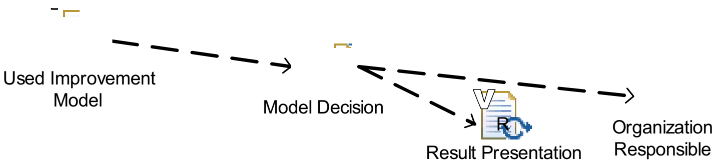


Figure 7: Set of related variants using the new proposed notation and its dependences

Furthermore, most of the variants and variation points in a process line are interrelated by means dependences. This means that a graph is created over the process line. This graph may be unconnected or cyclical, according to the dependences defined. We therefore know that by starting from each root variant it is possible to define a tree of variants which contains the variants the root type depends on by following dependences.

The definition of root and non-root variants is permitted by using the *VariantType* abstract class and two concrete classes associated with the variant class to model the type of variant. Figure 8 shows variant types related to the Variant abstract class, and how concrete root and non-root variants are defined by using the example of activity variants (*VActivity*).

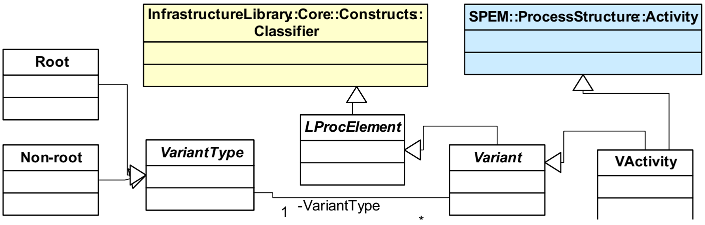

InfrastructureLibrary::Core::Constructs::Classifier
SPEM::ProcessStructure::Activity
Root
Non-root
VariantType
LProcElement
Variant
VActivity
1
-VariantType
*

Figure 8: Metamodel of variants including the specification of type.

4.2 Variation points

Variation points are empty points that the process line designer places in the core process to be occupied by different variants, which permits variability. However, processes are large, complex structures; they include several elements and the presence of dependences signifies that the realization of a single variation may imply several variants. Core processes therefore need a greater capability to be varied and to create consistent processes than the explicitly modelled variation points offer.

Two types of variation points can thus be distinguished:

- Explicit variation points. These are the points which are explicitly placed in the core process by the process line designer.
- Implicit variation points. These points appear in the core process exclusively as a result of the definition of the process itself. They are not explicitly defined by the process line engineer.

An example of an implicit variation point might be the set of roles working in a process. This point may be occupied by a role variant. The occupation rules in implicit variation points which ensure consistence in a generated process are shown in Sect. 4.3. Furthermore, a variant may depend on a variation point, as is explained in Sect. 4. However these variation points must be explicit variation points.

Since implicit variation points are not explicitly included in the process model, they may sometimes not appear in the model. But when they are occupied they must be included in the model. Some new icons have been defined to represent both implicit and explicit variation points with the aim of making diagrams more readable. Figure 9 shows an empty circle to represent explicit variation points and a dotted circle which represents an element that is an implicit variation point. Both implicit and explicit variation points are also shown, using these circles over the original icon.

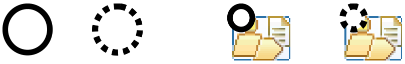


Figure 9: Circles to represent explicit and implicit variation points, and their application to activities

To allow us to distinguish between both types of variation points, Figure 10 shows the metamodel which portrays variation points as an abstract class associated with a *VariationPointType* class. This class can be defined as an Explicit or Implicit class, to represent the type of variation point. It can be used to define specific variation points, as is shown in the variation points activity (*VPActivity*).

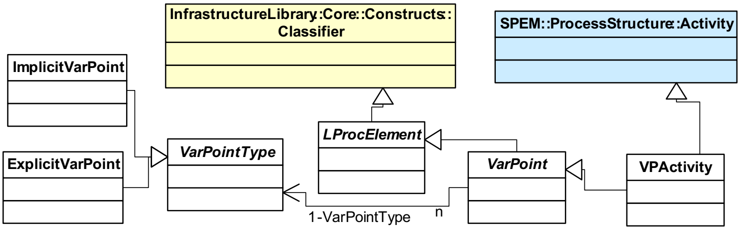


Figure 10: Metamodel of variation points with the specification of their type

4.3 Obtaining tailored processes from software process lines

Obtaining tailored process from a process line involves placing variants in variation points making use of variability. In Section 4.1 root and non-root variants have been defined, along with the implicit and explicit variation points defined in Section 4.2.

Moreover, to achieve correct tailoring it is important know why the variation is carried out. That is, what is the objective of carrying out that variation in the process. The objective is achieved by means of variations and is satisfied with the variants that the process engineer selects to vary the process. That is, from all the variants in the graph, the objective variants are those selected by the process engineer to vary the process. They themselves control the manner in which they and other variants are inserted into variation points. According the variants of Figure 7, we know that the Used Improvement Models root variant is also an objective variant, because it satisfies the “include the User Improvement Model in the Improvement Activity as input work product” objective variation in the core process of Figure 4-left. Other variants are part of the tree created from the objective variant because of their dependences.

Variants can be seen as root or non-root owing to the configuration and dependences of the process line, which implies the type can be seen as a state (Figure 8). However, variants are objective at the moment at which they are on the point of becoming part of the new tailored process through the occupation of a variation point. Objective variants can therefore be distinguished through the occupation relationship which links variants to variation points. The Occupation class in

Figure 3 thus adds the new *isObjective:boolean* attribute in order to distinguish the variants that satisfy objective variations.

As occurs with root variants, trees can be created by starting from objective variants and following dependences. These trees are composed of root and non-root variants. Used Improvement Models is the objective variant of the variants shown in Figure 7, and the remaining variants are therefore part of the tree it has created.

Furthermore, root variants and objective variants are related. Root variants may be included in a tree of objective variants for two reasons. The first is when the root variant is selected as an objective variant, as occurs in Figure 7. The second is when it is pre-selected as a default variant of an explicit variation point (and related because of a *VariationPoint-to-Variant* dependence). Non-root variants are included in this tree because the root variant depends on it. However, the process engineer may also select a non-root variant as objective variant. In this case, it is considered to be a root variant, its variability dependences as a supplier element are not considered and it defines its own tree.

When objective variation trees are going to be inserted in the core process the occupation of variants in variation points is limited owing to consistence. Variants include the *canOccupy (v:VariationPoint)* procedure shown in Figure 2. This procedure, which is defined for explicit variation points, ensures that variants are placed in suitable variation points because of consistence. As objective variants are selected to vary the process, they must satisfy the *canOccupy* procedure with the explicit variation points they are going to occupy. As regards root variants, if they are an objective variant they must satisfy the *canOccupy* procedure with an explicit variation point. If they are not an objective variant they are included in the tree because they are the default variant of an explicit variation point. Both reasons signify that the root variant must be placed in an explicit variation point and this point will be suitable for the variant. In the case of non-root variants acting as objective variants, they must occupy an explicit variation point as is explained with root variants.

However if non-root variants are not the objective of a variation they are considered in the core process owing to a root variant. There is therefore consistence between both variants as a result of dependences, and there may of course be consistence between the root variant and the explicit variation point of the core process (Figure 11). The consistence of non-root variants is thus ensured through the root variant. This means that it is unnecessary to check the consistency of the points they are going to occupy, and they can of course occupy implicit variation points, or explicit variation points. Table 1 summarizes the variants that can occupy different variation points.

Consistence between them ensured by means Variant2Variant Dependencies
Core Process

Figure 11: Ensuring consistence between non-root variants and implicit variation points by means of root variants

As we have seen, inserting the variant tree of Figure 7 into the core process of Figure 4-left is not possible because it does not have a role variation point, as was explained in Section 3.1. However, by using the proposed approach, this situation does not occur, since the role in this tree of variants is a non-root role variant and is not the objective variant. It can therefore occupy any of the implicit role variation points in the process, according to Table 1. The set of variants can be thus considered in the core process and it is possible to carry out the proposed variation, as is shown in Figure 11

To sum up, non-root variants are able to occupy implicit variation points. This makes it easy to ensure consistence in tailored process models, because it is possible to place the variants that are not the objective of a variation in generic places of core processes (named implicit variation points). Variants may thus be improved, or can be reused from similar process lines, and will be suitable for use.

Table 1: Occupation rules according to variants and variation point types

Type of variant	Type of variation point it may occupy
Root variant	Explicit variation points
Non-root variant but objective variant	Explicit variation points
Other non-root variants	Either explicit and implicit variation points

5 Conclusions

This work shows that it is necessary to tailor processes to make them usable. A variability mechanism based on variants and variation points additionally makes tailoring possible. These elements are therefore added to SPEM in order to enable process lines and tailored processes to be created. A classification of both variants and variation points, have been presented according to the dependences that are defined between them and their behaviour. Furthermore, we have analyzed how they are used to configure tailored processes and thus how their type affects the way in which they may interact with each other, showing an example of their application.

Overall, this approach facilitates the user’s capacity to personalize his/her own processes according to his/her necessities. Tailoring is simplified to the decision of which variants will satisfy the objective variations, and the selection of the variation points they are going to occupy. Moreover, process consistence is automatically ensured in a (from the user’s point of view) transparent manner by controlling the behaviour of the elements and managing related variants in the correct way.

This approach is also applicable to process modelling tools to allow them to model process lines, and enables processes to be automatically personalized. What is more important, since variants are atomic elements they can, if necessary, be readjusted during the process execution, and their use can be studied to improve the way in which they are used to improve processes, or to create new variants according to new variability necessities.

Furthermore, considering that variants are organization assets, they can be use to tailor different core process. Other variants can be created in order to better tailor processes. Both variants and core processes can be improved in their own way. The use of the presented approach to configure processes signifies that variants can be used to tailor processes throughout their evolution and improvement.

Future work is focused on three directions. The first is to carry out experiments to prove the understandability of this approach, and its application to real process families will later take place. Its application in the COMPETISOFT [24] process model will be carried out to adapt the model to several organizations in various Latin American countries. The second is to include this approach in process modeling tools with the aim of supporting process line management and process tailoring.

Finally, we are working on process institutionalization environment [8] which will include this approach and offers other capabilities such as the flexibilization of process models during their execution to make them similar to real processes. Furthermore, this makes the realization of process mining over variants and core processes possible by using the resulting executed processes to improve the variants, the core processes and thus the way in which processes are tailored.

Acknowledgments. This work is partially supported by the research into Software Process Lines sponsored by Sistemas Técnicos de Loterías del Estado S.A. within the framework of the agreement of the Innovación del Entorno Metodológico de Desarrollo y Mantenimiento de Software, the Program FPU of the Spanish MICINN, and by the ESFINGE (financed by the Spanish MICINN, TIN2006-15175-C05-05) and INGENIO (financed by the JCCM, PAC08-0154-9262) projects.

6 Literature

1. Clements, P. and L. Northrop, Software Product Lines. Practices and Patterns. 2002, Boston: Addison-Wesley.
2. Osterweil, L., Software Processes Are Software Too, in Proceedings of the 9th International Conference on Software Engineering. 1987, IEEE Computer Press: Monterey, CA. p. 2-13.
3. Rombach, D. Integrated Software Process and Product Lines. in International Software Process Workshop 2005, LNCS 3840. 2005. Teddington, UK: Springer-Verlag.
4. Humphrey, W., Managing the Software Process. 1989: Addison-Wesley.
5. Yoon, I.-C., S.-Y. Min, and D.-H. Bae, Tailoring and Verifying Software Process, in 8th APSEC. 2001. p. 202-209.
6. Ocampo, A., F. Bella, and J. Münch, Software Process Commonality Analysis. Software Process - Improvement and Practice, 2005. 10(3): p. 273-285.
7. Chrissis, M.B., M. Konrad, and S. Shrum, CMMI: guidelines for process integration and product improvement. The SEI series in software engineering. Vol. 1. 2006, Boston: Pearson. 676.
8. Martínez-Ruiz, T., F. García, and M. Piattini. Process Institutionalization using Software Process Lines. in to be published in ICEIS. 2009. Milan.
9. Martínez-Ruiz, T., F. García, and M. Piattini. Towards a SPEM v2.0 Extension to Define Process Lines Variability Mechanisms. in Software Engineering Research, Management & Applications. 2008. Praga: Springer Verlag.
10. Denger, C., F. Elberzhager, and T. Schulz, Customization Approach for Inspection considering Influence and Variation Factors, in BMF Project, F. IESE, Editor. 2008, Fraunhoufer IESE: Kaiserslautern.
11. Ocampo, A. and J. Münch, Software Process Variability: Concepts and Approaches. 2004, Institute for Experimental Software Engineering, Kaiserslautern.
12. Simidchieva, B.I., L.A. Clarke, and L. Osterweil, Representing Process Variation with a Process Family, in ICSP 2007, Q. Wang, D. Pfahl, and D.M. Raffo, Editors. 2007, Springer-Verlag. p. 121-133.
13. Bayer, J., M. Kose, and A. Ocampo, Improving the Development of e-Business Systems by Introducing Process-Based Software Product Lines, in 7th PROFES. 2006, Springer-Verlag: Amsterdam. p. 348-361.
14. Lu, R. and S. Sadiq, A Reference Architecture for Managing Business Process Variants. Proc. 9th International Conference on Enterprise Information Systems (ICEIS2007), 2007.
15. Clauss, M. Generic Modeling using UML extensions for variability. in Workshop on domain specific visual languages. 2001. Tampa Bay, Florida.
16. Webber, D. and H. Gomaa, Modeling Variability in Software Product Lines with the Variation Point Model. Science of Computer Programming, 2004. 53(3): p. 305-331.
17. Schmid, K. and I. John, A customizable approach to full lifecycle variability management. Science of Computer Programming, 2004. 53(3): p. 259-284.
18. Schneiders, A. and M. Weske, Activity Diagram Based Process Family Architectures for Enterprise Application Families, in In Enterprise Interoperability: New Challenges and Approaches, Springer-Verlag, Editor. 2007: London. p. 67-76.
19. Asikainen, T., T. Männistö, and T. Soimimen, Kumbang: A domain ontology for modeling variability in software product families. Advanced Engineering Informatics, 2007. 21(1): p. 23-40.
20. Sinnema, M. and S. Deelstra, Industrial Validation of COVAMOF. Journal of Systems and Software, 2007. 49(1): p. 717-739.
21. Goedicke, M., C. Köllmann, and U. Zdun, Designing runtime variation points in product line architectures: three cases. Science of Computer Programming, 2004. 53(3): p. 353-380.
22. OMG, Software Process Engineering Metamodel Specification. 2007, Object Management Group.

23. Sinnema, M., et al., COVAMOF: A Framework for Modeling Variability in Software Product Families. SPLC 2004, 2004(Springer Verlag): p. 197-213.

24. Oktaba, H., et al., COMPETISOFT: A Improvement Strategy for Small Latin-American Software Organizations, in Software Process Improvement for Small and Medium Enterprises: Techniques and Case Studies, H. Oktaba and M. Piattini, Editors. 2008, Idea Group Inc.

7 Author CVs

Tomás Martínez-Ruiz

Tomás Martínez is a MSc of Computer Science at the University of Castile-La Mancha (UCLM). He is a PdH student at the ALARCOS Research Group of that University. His research interests are software process improvement, software process lines and aspect-oriented software. Contact details are: Escuela Superior de Informática, Paseo de la Universidad 4, 13071-Ciudad Real, Spain; tomas.martinez@uclm.es.

Félix García

Dr. Félix García is an Associate Professor at the University of Castilla-La Mancha (UCLM). His research interests include business process management, software processes, software measurement and agile methods. He holds MSc and PhD degrees of the UCLM in Computer Science, and is a member of the ALARCOS Research Group of that University, specialized in Information Systems, Databases and Software Engineering. Contact details: Escuela Superior de Informática, Paseo de la Universidad 4, 13071-Ciudad Real, Spain; felix.garcia@uclm.es.

Mario Piattini

Dr. Mario Piattini has an MSc. and PhD in Computer Science from the Technical University of Madrid and MsC in Psychology from the UNED, and is a CISA, CISM and CGEIT by ISACA (Information System Audit and Control Association). CSQE by ASQ. He is a professor in the Department of Computer Science at the University of Castilla-La Mancha, in Ciudad Real, Spain. Author of several books and papers on software engineering, databases and information systems, he leads the ALARCOS research group of the Department of Information Systems and Technologies at the University of Castilla-La Mancha. His research interests are: software process improvement, software metrics, software maintenance and security in information systems. Contact details: Escuela Superior de Informática, Paseo de la Universidad 4, 13071-Ciudad Real, Spain; mario.piattini@uclm.es

Survey of Practitioners' Attitudes To Software Process Modeling

Tuomas Mäkilä, University of Turku, Finland
Henrik Terävä, Digia Plc., Finland

Abstract

Software process modeling has evolved fast during the past few years. New dedicated modeling standards and process-based tools have been introduced. Emerging trends of the process modeling could bring even more radical changes. These changes have affected the work of common software industry practitioners. Results of a qualitative survey, which was conducted amongst Finnish software practitioners, are presented in this paper. The goal of the survey was to map the attitudes of the practitioners towards the use of software methodologies and software process modeling in their own work. In addition the practitioners provide expert analysis on the emerging modeling trends. The answers of the practitioners are analyzed in this paper and conclusions are given on how the practitioners see the current state and the near future of the software process modeling.

Keywords

Software Process Modeling

1 Introduction

Software process modeling and software process models are essential techniques in making understandable process descriptions. Efficient modeling and clear, up-to-date process models are essential part of many software process improvement (SPI) activities. Several different techniques have been used to construct the models: Work-flow diagrams, more advanced business process modeling languages, and also dedicated software process modeling languages. Development in the field of software process modeling has enabled new applications for the models and brought the models closer to the every-day project work in the software industry.

Many practical applications of the process modeling require strong tool support. For example a simple task like keeping a process model up to date can become laborious with basic office tools. Existing modeling language standards simplify and in many cases enable the implementation of modeling tools. Modeling languages standards are also essential for model reuse and model interchange between different organizations. There has been swift development in both modeling standards and tool support during the past five years. This development has increased the software industry’s interest towards the software process modeling.

Second version of the Software Process Engineering Metamodel (SPEM) modeling language standard [1] was released in 2007. The SPEM modeling language has provided a foundation for development of new generation of software process modeling tools. IBM has constantly published improved versions of its Rational Method Composer (RMC) [2] modeling tool which is indirectly based on the SPEM standard. Partly based on the RMC code, almost identical tool is freely available through the Eclipse Process Framework (EPF) project [3]. The EPF project also distributes models of several popular process methodologies. Latest advancement is the IBM's Team Concert tool [4] which uses simple process models as a mean to configure the project tools (e.g. user rights, version tracking rules and communication). Microsoft has also same kind of ideas in their Visual Team System product [5].

This paper reports the results of a qualitative survey of software practitioners' attitudes on software process modeling. There were three goals for the survey. First goal was to get objective information about the current state of the process modeling in the software industry. This was done by obtaining information about practitioners’ personal attitudes towards process modeling and also about the underlying work methodologies. Second goal was to get expert analyses on the upcoming trends of the process modeling. Third goal was to evaluate the research team’s own beliefs of the process modeling which were based on the literature and the team’s own experiences.

The research was conducted as a two-part qualitative survey. In the first part the respondents answered the web-based questionnaire at their own pace. The questionnaire included mainly open questions grouped in three categories: 1) Influence of work processes in respondents' own work, 2) influence of process models in respondents' own work, and 3) analysis on the upcoming trends of process modeling. In the second part of the survey the answers of the selected respondents were supplemented during personal interviews. The interviews were done after the preliminary analysis of the answers of the first part questionnaire.

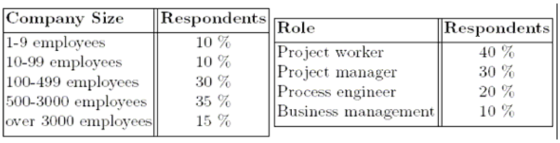
The survey results were analyzed by our research team. The team consisted of both university researchers and industry practitioners. The idea was to use a scientific method to get relevant results, which would also serve industrial needs. During the analysis the research team tried to find common elements and also contradictions in the answers to form meaningful conclusions. In addition to the qualitative analysis, the team used straightforward quantitative methods to analyze the respondents' answers to certain, individual questions. It should be noted that the percentage values presented in this paper can not be directly generalized outside this study.

The structure of the paper is following. In Section 2 the respondents' opinions on the impact of the software methodologies, processes and models to their every-day work are presented. In Section 3 respondents' views on the selected software process modeling trends and the future of the software process modeling are analyzed. Section 4 presents the research team’s plans on continuing and extending the research of practitioners' opinions on the software process modeling. Finally in Section 5 the paper is concluded.

2 Current State of Modeling

The questionnaire was send to several software companies from the research team's partner network in Finland during spring 2009. Twenty practitioners (N=20) from fifteen different companies participated the survey. The total number of people who actually received the invitation to the survey is hard to determine, but about one fourth of those who opened the survey web page actually filled the survey.

Respondents' company sizes varied quite evenly from micro companies to large companies. The roles of the survey respondents varied from developer and project manager to process engineer and company executives. When the more detailed work descriptions of the respondents were analyzed there was usual variation between the work descriptions. No particular sector of the ICT industry was overemphasized. Most of the respondents worked in the various software development projects, but there were also respondents working in e.g. ERP system development, methodology engineering and maintenance projects. The exact distributions of company sizes and respondent work roles are presented in Table 1.



Company Size	Respondents
1-9 employees	10 %
10-99 employees	10 %
100-499 employees	30 %
500-3000 employees	35 %
over 3000 employees	15 %

Role	Respondents
Project worker	40 %
Project manager	30 %
Process engineer	20 %
Business management	10 %

Table 1 Company sizes and roles of the respondents

The respondents were inquired about the methodologies and process frameworks their employees had implemented. This was done to better understand the use of process modeling and modeling needs in respondents' everyday work. Based on the detailed answers, it can be said that most of the respondents were quite well aware of the methodologies used in their companies and the maturity of the used methodologies.

It was found out that 85 % of the respondents recognized at least one methodology to be used in their company. The methodologies used in the respondents' companies varied and there was not one clearly dominant methodology. For example ISO standards, ITIL, RUP or variant, CMM(I) and Agile methodologies were mentioned by multiple respondents. Half of the companies used more than one methodology. Usually a standard methodology was accompanied by company's own process methodology or guidelines. The maturity of the used methodologies also varied: 40 % of the companies had used a methodology for several years and 40 % were at the beginning of the methodology adoption. It could be seen from the answers that almost all of the companies were constantly developing their processes and evaluating underlying methodologies.

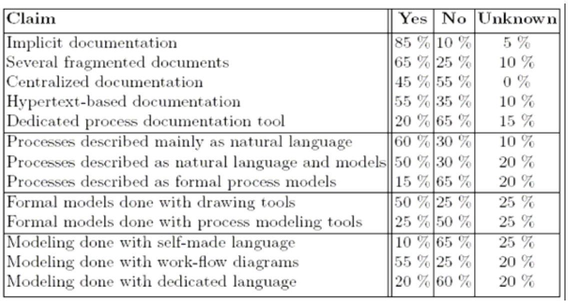
The information presented above is mainly background information. The respondents' views on the benefits and drawbacks of the methodologies and process modeling are analyzed next.

By analyzing the overall attitudes of respondents it can be concluded that 65 % of the respondents found the process frameworks beneficial for their work, 15 % had negative experiences, and 20 % had neutral attitude. Negative experiences seemed to be result of poorly defined or too inflexible processes. Although many respondents had positive overall experience, most of them also found some negative aspects in the use of the methodologies. In addition many respondents emphasized that the use of the guidelines and the methodologies has to be adapted case by case in order to get the most out of them.

More detailed analysis of attitudes revealed that the many respondents saw the methodologies as a kind of foundation for either the development work itself or the improvement activities. Many of them also said that methodologies enable reuse of practices which in turn saves time in different phases of project work. On the other hand the reuse could lead to repeating old mistakes. Other more negative attributes connected to the methodologies were inflexibility and unnecessary overhead caused by a methodology.

Although only 20 % of the respondents worked mainly with methodology improvement issues, almost all respondents had participated in the process improvement activities: 55 % directly, 35 % indirectly e.g. by giving feedback about the process, and 10 % had not participated at all.

The use of the software process modeling was investigated by a set of yes / no claims about the state of the process descriptions and modeling in the respondents' companies. The answers are collected in Table 2. It can be said that while the use of standard methodologies was unexpectedly high in the respondents' companies the use of process modeling seemed to be more normative. The table shows that the advanced process modeling techniques like dedicated modeling tools, formal process models and use of modern process modeling language were still rare in the companies. It is also notable finding that number of the *"unknown"* answers increased when more technical issues were inquired.



Claim	Yes	No	Unknown
Implicit documentation	85 %	10 %	5 %
Several fragmented documents	65 %	25 %	10 %
Centralized documentation	45 %	55 %	0 %
Hypertext-based documentation	55 %	35 %	10 %
Dedicated process documentation tool	20 %	65 %	15 %
Processes described mainly as natural language	60 %	30 %	10 %
Processes described as natural language and models	50 %	30 %	20 %
Processes described as formal process models	15 %	65 %	20 %
Formal models done with drawing tools	50 %	25 %	25 %
Formal models done with process modeling tools	25 %	50 %	25 %
Modeling done with self-made language	10 %	65 %	25 %
Modeling done with work-flow diagrams	55 %	25 %	20 %
Modeling done with dedicated language	20 %	60 %	20 %

Table 2 How the work processes are documented in respondents' companies

The process modeling tools and languages the respondents use in their work was also inquired. 35 % of the respondents used dedicated process modeling tools, 25 % used some other tools like drawing software while 40 % did not use any process modeling tool or language. SPEM was mostly used modeling language while Business Process Modeling Notation (BPMN) and Universal Modeling Language (UML) were also in use.

The concrete use of the process models revealed that about half of the respondents used models just to access common document templates or checklist. Only one fourth of respondents mentioned process models as a tool for process tailoring and software process improvement. The question about the use of process models also showed that many people used the word *"model"* as a synonym for the word *"methodology"*. This was confusing because the research team used the word *"model"* to represent a result of modeling efforts.

The respondents would develop the use of the process models and modeling in their company in various ways. Only 75 % of all respondents actually answered to the model development question. All of them saw improvement possibilities and were able to specify clear development suggestions. Many respondents mentioned that the modeling tools should be improved and the use of dedicated modeling language increased. One respondent noted that this could take longer than expected: *"Formal modeling is our next step but this step is bigger than we first thought"*.

Some respondents wanted to increase flexibility of the models. One rationale behind this was to enable tailoring of the models for different situations. There were also suggestions about making a library of more detailed process models for different small scale situations. This approach resembles the emerging practice-based process modeling [6] [7] which is probably still quite unknown amongst the practitioners. Rest of the respondents wanted to increase the use of models by making them clearer, easier to read, and more comprehensive. Also training for using the models was needed.

Answers about the process modeling seemed to suggest that the maturity of the software process modeling was lower than the maturity of software methodologies which were used in companies. 85 % of the respondents knew that their company utilizes at least one development methodology but only 60 % used some kind of modeling tool. It also seems that the respondents were more familiar with methodologies and their use than meaning and the use of the software process modeling.

3 Emerging Trends

The second goal of the survey was to get practitioners' analysis on the vitality of emerging software process modeling trends. Three most interesting modeling trends were selected based on the research team's previous research [8] and the recent advancements in the modeling techniques. The trends that were presented to the respondents were:

- **Distributed process modeling.** Distributed process modeling is an approach which emphasizes bottom-up modeling practices. Portions of the process models are done in the projects where the process is used. This approach is opposite to the top-down, process engineer led process modeling. In practice, both aspects have to be taken into consideration. Techniques for distributed process modeling are proposed e.g. in the paper [9].
- **Light-weight process modeling.** Light-weight process modeling means focusing only on the most important elements during the process modeling. Idea is to quickly form a starting point for longer modeling efforts or to quickly illustrate the current state of the process. The approach is opposite to the traditional business modeling techniques where target is to generate very accurate models. Techniques for light-weight process modeling are described in the papers [10] [11].
- **Decrease of project-process-gap.** There's always overhead when process description and methodologies are enacted into an actual project organization. The gap can lead to process deviations and make measuring the project difficult. The gap can be decreased with modern process modeling techniques for example by configuring the project tools using the actual process descriptions. Ivar Jacobson has discussed about the project-process gap and developed a practice-based method to deal with the issue [6]. Process modeling techniques for reducing the gap are presented e.g. in the paper [12].

The respondents were asked to evaluate whether the techniques described in the trends would be applicable to their own work and if they would benefit from these trends. They were also inquired to analyze if some of the techniques were already applied in their companies.

Surprisingly many of the respondents had hands-on experiences on *the distributed process modeling*: 30 % of them had at least tested the principles related to this trend. Half of the respondents had positive attitudes towards this trend while others were neutral about the trend. The respondents liked the idea that the process users can directly affect the process models. They also saw that it is efficient to define process where it is used. This way there would be less process deviations in the project level and the overall process model would better resemble the reality.

The respondents found also many possible pitfalls in the distributed process modeling. The largest concern was the integration of the distributed models into one company-wide model. Many respondents mentioned that strict distributed modeling would not work, but traditional top-down techniques would still be needed to accompany the distributed modeling. There were worries about extra workload and insufficient skill levels of the project workers who would have to participate more actively in process modeling. As a solution the respondents offered a modeling facilitator who would do the actual modeling in co-operation with the project team, and take care of the integration and other technical issues.

The light-weight process modeling was a little bit more unfamiliar concept: Only two of the respondents had tried the techniques related to this trend. Controversially to the previous trend even 70 % of the respondents had positive attitudes towards this trend. The respondents liked the idea to model only necessary elements and reduce unnecessary overhead. The trend was connected to the principles of agile methods by several respondents. Iterative process development was also mentioned.

The largest problem with light-weight modeling was how to identify the most important process elements and define the detail level of the modeling. Solutions for this problem were not found. Some respondents also identified that the light-weight process modeling has very focused applicability: It works best for sketching and piloting new methodological ideas, and for forming a starting point for longer lasting process modeling efforts.

The decrease of the project-process gap with process modeling techniques was clearly the most abstract concept for the respondents. Although 65 % of the respondents had positive attitudes towards this trend, the analysis was not as detailed as with the previous trends. The main message was that it is hard to see how the project-process gap is actually decreased because the tools and modeling standards do not yet fully support this approach.

The survey was concluded with the question about the respondents' own opinions on the future trends of the software process modeling. Almost all respondents saw that meaning of the process modeling and process methodologies will generally increase in the near future. The respondents emphasized the importance of the development of the both process and project tools, and their interoperability. It seemed that there are methodologies, modeling languages and tools available already but their maturity is still quite low. Full potential of the modeling technologies is still to be reclaimed. Optimistically, the respondents believed that this will eventually happen.

4 Further Work

As mentioned before, this was a qualitative survey and therefore percentages presented in the paper give only hints on which issues were more important and which were less important for the respondents as a whole. Because of the small sample (N=20) the percentage values themselves are not statistically significant.

This study acts as a starting point for a longer research on the practitioners' attitudes and expectations towards the software process modeling. The qualitative analysis presented in the paper was a necessary step to form understanding of the modeling issues that are important for the practitioners. Next the research team is planning to conduct a statistical survey that will provide more comprehensive information on tighter formulated set of hypotheses.

In the study presented in the paper the population and the sample was limited to the Finnish software practitioners. Therefore conclusions can be drawn only about Finnish software industry. In the following study the survey will be conducted in several other countries as well. There might be regional differences on the attitudes since the software methodology culture seems to vary geographically. In the following study the research team expects to deepen understanding on the differences of the attitudes of different employee groups by using more formal statistical analysis.

It will also be interesting to follow how the field of the software process modeling will evolve in the near future. As the study continues the research team will observe how well the expert analyses of the respondents realize.

5 Conclusions

There should not be big surprises in the overall results of the survey for those who have followed the recent development of the software process modeling concepts, methods, languages, and tools. Practitioners welcome, with healthy criticality, new methodologies that will improve their ability to do their every-day work better. Naturally the methodological frameworks do not offer a silver bullet, but some kind of structures and guidelines are clearly needed in the software work.

The software process modeling concepts seem to be still a little bit unfamiliar for the practitioners, although the project and development methodologies are quite well known. Reason for this might be the immaturity of the modeling languages and the tools. It should also be noted that only portion of the practitioners actually modify the process models, and therefore work directly with the modeling tools and languages. For others it is sufficient to understand the models and maybe give constructive feedback about them. This situation might however be changing because of the trends presented in this paper.

The unity of the practitioners' answers for the survey was interesting. Despite the fact that the respondents represented many different work roles and different-sized companies, they all looked quite positively at the methodological issues and changes in the process modeling field. Maybe the people who are interested in these kinds of issues became selected as the respondents and this somehow biased the results. However, it can be said that there are people in the software industry who are openmindedly willing to adopt new methodologies, but at the same time they expect to see direct improvements in their work environment.

Acknowledgements: The authors wish to thank all practitioners who participated the survey from Digia, Codebakers, Ericsson, RP5 Software, Samlink, Sesca and several other software companies.

Literature

[1] Object Management Group. Software & systems process engineering meta-model specification - version 2.0, April 2008.

[2] Rational method composer (IBM) homepage. http://www-01.ibm.com/software/awdtools/rmc/. Accessed on May 14th 2009.

[3] Eclipse process framework (EPF) project homepage. http://www.eclipse.org/epf/tool component/tool index.php. Accessed on May 14th.

[4] Rational team concert homepage. http://www-01.ibm.com/software/awdtools/rtc/. Accessed on May 14th 2009.

[5] Team system (Microsoft) homepage. http://msdn.microsoft.com/enus/teamsystem/default.aspx. Accessed on May 14th 2009.

[6] Ivar Jacobson, Pan Wei Ng, and Ian Spence. Enough of processes: Let's do practices. Journal of Object Technology, 6(6):pp. 41-66, August 2007.

[7] IBM practices homepage. http://www.ibm.com/developerworks/rational/practices/. Accessed on May 14th 2009.

[8] Antero Järvi, Tuomas Mäkilä, and Harri Hakonen. Changing Role of SPI Opportunities and Challenges of Process Modeling, volume 4257/2006 of Lecture Notes in Computer Science, pages 135-146. Springer Berlin / Heidelberg, 2006.

[9] Oktay Turetken and Onur Demirors. Process modeling by process owners: A decentralized approach. Software Process: Improvement and Practice, 13(1):75-87, 2008.

[10] Tuomas Mäkilä, Antero Järvi, and Luka Milovanov. Light-weight approach for software process modeling - a case study. In Proceedings of New Exploratory Technologies 2007, October 2007.

[11] Paula Savolainen, Hanna-Miina Sihvonen, and Jarmo Ahonen. SPI with Lightweight Software Process Modeling in a Small Software Company, pages 71-81. 2007.

[12] Soojin Park, Hoyoung Na, Sooyong Park, and Vijayan Sugumaran. A semiautomated filtering technique for software process tailoring using neural network. Expert Systems with Applications, 30(2):179-189, February 2006.

6 Author CVs

Tuomas Mäkilä (tuomas.makila@utu.fi)

M.Sc.(in technology) Tuomas Mäkilä works as a teacher and a resercher at the Department of Information Technology of the University of Turku. He has researched software process modeling for five years and is currently finishing his doctoral thesis on the topic.

Henrik Terävä (henrik.terava@digia.com)

M.Sc.(in technology) Henrik Terävä works as a project manager at Digia Plc., a worldwide information and technology solutions provider. He is an expert on modern software process modeling technologies and develops Digia's Open Method software development methodology product.

Approaching a software factory process philosophy for a SME in Spain

Luis Fernández-Sanz, José R. Hilera, José J. Martínez, Guillermo López, Jesús Serrato

Abstract

ADV is a Spanish Small Medium Enterprise (SME) software manufacturer interested in improving processes to approach software factory philosophy for projects. As a result of a joint project with Universidad de Alcalá, a new process for automating code production has been devised following a component-based approach. After applying this approach to production of COTS software, benefits reported by developers suggest a significant improvement in maintainability and quality as well as in cost reduction although adaptation and training requires medium-high level of effort.

Keywords

Software factory, SME, component-based, COTS

1 Introduction

ADV is software manufacturing SME located at the Technology Entrepreneurship Area of Alcalá de Henares (Spain). Software products of ADV are used by more than 400 companies covering different areas as digital printing companies (57%), cardboard manufacturers (11%) and a great variety of customers, none of them represents more than 2% of the business. It usually collaborates through R&D projects with Universidad de Alcalá (UAH) in the area of software development and engineering. In 2007 it launched a project to implement an improved process for code generation for environments based on Visual Studio, .Net and Postgres Database. The idea was to apply to code generation and test a philosophy equivalent to software factories.

Obviously, the term software factory embraces a variety of concepts not always equivalent. On one side, Microsoft has included this expression to its .net framework while promoting a fast component integration scheme with a limited conceptual framework. In fact, [4] defines a software factory as production approach where tools, processes and contents are configured to allow automation of development and maintenance of archetypical products, adapting, assembling and configuring components of its MS Framework. However, during 21st century, many software factories initiatives conceived as work centres promoted by large multinational companies have been implemented. In many cases, the idea was centred in specific types of development projects and the emphasis was on adequate location (off-shore, near-shore, close-shore, etc.) to promote cost savings (e.g., salaries, installation costs, taxes, etc.) where life cycle was limited to detailed design, code generation and testing; project teams were permanent and using always the same routines and processes. Adopting this view, definitions as “high productivity environment for software development to achieve competitive solutions in time and cost for the customers while following software development standards” or “service unit where each group of specialist is responsible of a specific phase of life cycle: definition, specification, development, testing and implementation”, etc.[5] are applicable to these initiatives. In general, high level of organization and coordination, mature processes, management disciplines and automation with tools are common characteristics [6].

ADV decided to work along this line of specialized work on code generation for developing specific products through intensive use of tools and component-based process. Best practices extracted from traditional process models were implemented for initial phases of life cycle, especially on requirements management. In order to address this challenge, a specific light-weight process supported by tools was created as recommended in [4][7]. The basic idea is productivity enhancement through a component-based philosophy [8][9] with strong tool support.

In this paper, Section 2 presents implemented life cycle and development processes while section 3 is focused on process automation. Section 4 presents some results from the implementation of the process and Section 5 outlines conclusions and future lines of work.

2 Process for design and code generation based on components

The R&D experts (2 researchers) and the development team (2 developers) of UAH began the project with a series of meetings with managers of ADV as well as subsequent proposals of code generation processes to optimized it to the specific characteristics of applications portfolio of the company. The result, after several versions and test, was a sequence of four activities which has been implemented as guideline for detailed design and code generation of each application starting from a data model as seen in Figure 1. Defined processes and activities are the following ones:

- **Process 1, Data model and data base definition**. Starting from specifications a description is generated as a component which may be reused for application which is based on the same model. It is also possible to get this definition from an existing database.
- **Process 2, Generation of data components**. Adopting a component-based approach within a mutilayer architecture, next step is the generation of control classes of relations between applica-

tion and database. According to the proposed methodology, code generation is supported by common components which are independent of specific data base to be chosen; they will be complemented by those generated in step 1.

- **Process 3, User interface generation**. Graphical user interface components for the presentation layer will be generated from the components generated in previous phases. Again, a series of basic components are available for the organization but developers may create more advanced ones combining basic elements. It is also possible to combine predefined styles to components to enable customization.
- **Process 4, Application composition**. Last stage (consistent with traditional component methodologies [8][9]) is the composition and integration of components. Here many of the advantages of following previous steps arise because developers do not need to dive in the code to generate the application. Thanks to supporting mechanism based on tools, a step-by-step process is devised and code is automatically generated. Anyway, developers have the option of changing external view of the application because several global styles may be applied to components. EuroSPI Listings.

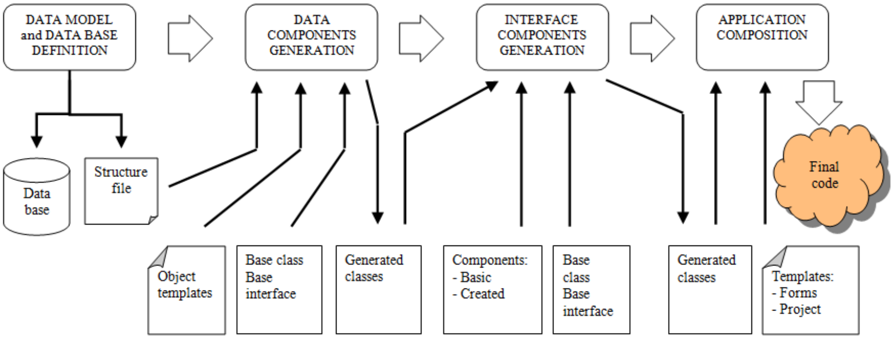
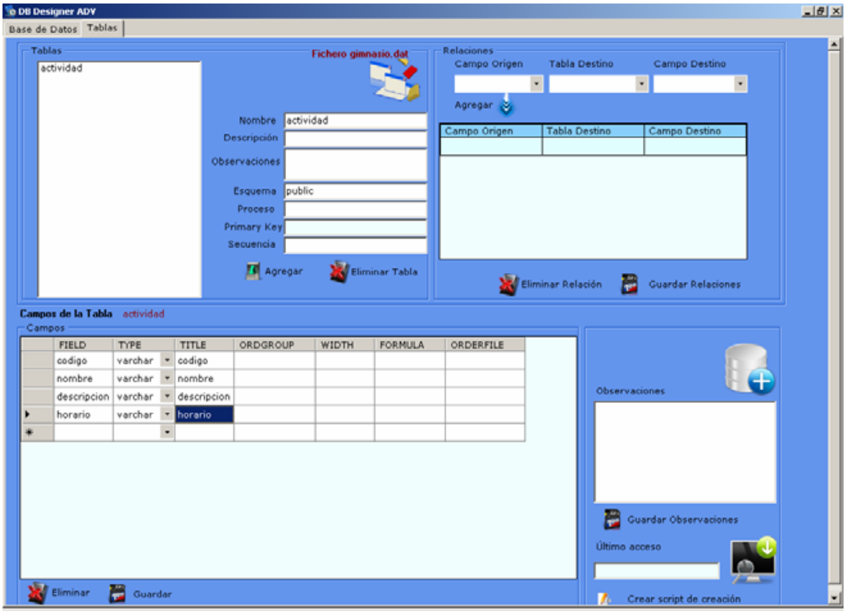


Fig. 1. Implemented process for code generation

3 Automation details

This component-based process for code generation adapted to a portfolio of business applications has been implemented using several tool complements added to Visual Studio environment. This enables a compact support to development where it is possible to integrate controlling and management mechanisms like configuration management of components and elements with the subsequent supporting actions. Development size with the corresponding supporting libraries was 30 KLOC. Major difficulties arose in the integration with Visual Studio due to some inconsistencies in the documentation regarding its real behaviour.

In general, all basic steps integrate Wizard-like helpers to assist developers. This enables a good standardization of code as well as better results during testing given the fact components have been exhaustively tested. All these features lead to easier maintenance.



Tablas: actividad

Nombre: actividad

Esquema: public

Campos de la Tabla actividad

FIELD	TYPE	TITLE	ORDGROUP	WIDTH	FORMULA	ORDERFILE
codigo	varchar	codigo				
nombre	varchar	nombre				
descripcion	varchar	descripcion				
horario	varchar	horario				

Fig. 2. View of one of the components of the supporting environment

4 Benefits perceived by developers

As presented above, this paper presents a project to improve productivity and quality based on implementing a new code generation process within the life cycle of a SME. Given the characteristics and environment of the company, it was decided to intervene in the smallest portion of development activities to generate value for developers with a strong tool support integrated within other process like configuration management and project management.

In order to analyze impact of this initiative, developers of the company who have experienced the integration of this set of process and tool environment in their daily project practice were requested to answer an anonymous questionnaire. The sample included programmers as well as project managers with an experience in software development ranging from 2 to 23 years (average of 8.1 years). Collected data are still limited because not all the technical workforce of the company has experienced the new methods in real work although 7 and internal and 7 external workers have answered the survey.

Results can be summarized as follows:

- 100% of respondents report reduction in effort to generate code for applications. A 54% reduction is the average of their estimation. A basic application with a limited amount of objects and tables (e.g., classical basic library management system requires a bit more than two hours to have an almost final version).
- 71.4% considers this generation process have a strong impact on getting higher reliability of code whereas the others rank the improvement as light.

- 85.7% report a high enhancement of code maintainability whereas the rest suggest a light one. Improvement of maintainability is supported by the fact applications are based on a common structure and developers always work in the same level of abstraction.
- 57.14% of respondents indicate that new process and associated tool support requires a big effort of adaptation by developers: the other half ranks the required effort as medium. They reported an average effort of 4 days to reach an adequate level of command, of course once a minimum expertise in Visual Studio and .net is present.
- However, 100% accepts the new approach because benefits overcome the inconvenience of the adaptation phase.

Although these are not definitive data, management of the company as well as developers are satisfied with the results.

5 Conclusions

Although software process improvement proposals tend to be centred in general life cycle processes and known process models, for many SMEs their firsts steps to gain productivity as well as process maturity is to implement specific actions with not so conventional methodologies. In this case a software manufacturing SME which core business centred in COTS software for industrial decided to work on the line of component-based development because it would offer more solid benefits. Thanks to an R&D project, ADV is benefiting from a more mature code generation process fully integrated with a strong tool support to its main development environments based on .net and Visual Studio. It is also integrated with configuration management processes supported with a CVS system through an integration utility (WinMerge). ADV is not planning the implementation of SPI based on general models (CMMI [10] o ISO 15504 [11]) in a short term although it is included in its long-term strategy. However, informal evaluation of current processes of ADV done by managers suggest the company would cover practices associated to an equivalent to level 3 of staged representation of CMMi in different areas.

As an ongoing line of action, company is implementing a full automated monitoring and effort measurement process to feed data to its project management process thanks to the capture of Visual Studio events and its recording in local files as activity log data for developers. This data will help in the creation of more precise estimation methods for projects once a significant database with cumulated historical data, especially because the MS solution (Team System) is unaffordable by a SME. Anyway, UAH and company have agreed to collect more data on real usage and benefits through questionnaires and effort information captured from configuration management system and environment events: the objective is to analyze more precisely the impact of the solution

6 Literature

1. Laporte C.Y., Alexandre, S., O'Connor, R.: A Software Engineering Lifecycle Standard for Very Small Enterprises, In: Proceedings of the 15th Conferences on Software Process Improvement EUROSPI'08, pp 129-141 (2008).
2. Garcia, I., Calvo-Manzano, J., Cuevas, G., San Feliu T.: Determining Practice Achievement in Project Management Using a Two-Phase Questionnaire on Small and Medium Enterprises, In: Proceedings of the 14th Conferences on Software Process Improvement EUROSPI'07, pp. 46-58 (2007).
3. Chen, X., Staples, M.: In: Proceedings of the 14th Conferences on Software Process Improvement EUROSPI'07, pp. 59-70 (2007).
4. Greenfield, J., Short, K.: Software factories. Assembling applications with patterns, models, frameworks and tools. Wiley (2004)
5. Escribano, C.: 12 años de fábrica de software: lecciones aprendidas, In: First Software Factories International Conference, www.calidaddelsoftware.com (2007)

6. Rodríguez, A.: Software Factory: desarrollo de calidad para procesos de desarrollo a medida, In: First Software Factories International Conference, www.calidaddelsoftware.com (2007)
7. Fernstrom, C.; Narfelt, K.-H.; Ohlsson, L.: Software factory principles, architecture, and experiments, IEEE Software, 9-2, 36 – 44 (1992)
8. Heineman, G., Council, C.: Component-Based Software Engineering - Putting the Pieces Together. Addison-Wesley (2001).
9. Wallnau, K. Hissam, S., Seacord, R.: Building Systems from Commercial Components. Addison-Wesley (2002).
10. CMMI team: CMMI for Development Version 1.2, Software Engineering Institute (2006).
11. ISO/IEC 15504, Information technology: Process assessment, Parts 1-7, International Organization for Standardization (2008).

Acknowledgments

This work was supported by the R&D project UAH 50/2007.

7 Author CVs

Luis Fernández Sanz

Luis Fernandez_Sanz received his PhD in Computing from University of the Basque Country in 1997. He has over twenty years of experience in university teaching in Computing degrees in Universidad Politecnica de Madrid and Universidad Europea de Madrid. Now he is with Universidad de Alcala as assistant professor. His research interests include software quality and testing.

José Ramón Hilera González

PhD. from University of Alcala, José Ramón received a Telecommunications Engineer degree from Universidad Politécnica de Madrid. Associate professor in the Computer Science Department of the University of Alcalá. His research interests mainly focus on topics related to software and web engineering, e-learning and knowledge representation, where he supervises several PhD works in these areas. Author or co-author of more than 100 scientific works (books, articles, papers and research projects).

José Javier Martínez Herraiz

PhD. from University of Alcala, José Javier received a Computing Engineer degree from Universidad Politécnica de Madrid. He is the Director or the Computer Science Department of the University of Alcalá. His research interests are mainly related to software engineering, information engineering and e-learning, where he supervises several PhD works in these areas.

Guillermo López

Guillermo is the CEO of the company ADV. With more than 30 years of experience in software development and management, his research interests include software development productivity and quality.

Jesús Serrato

Jesus received a Computing Engineering degree Universidad Politécnica de Madrid. He is the head of the software factory in the company ADV (Spain) since 1988.

Successful Process Improvement for Small and Medium Sized set ups

Kurt S. Frederichsen
Center for Software Innovation
Alsion 2, DK-6400 Sønderborg, Denmark
ksf@cfsi.dk
www.cfsi.dk (in Danish)

Abstract

In this paper, we describe a framework with three perspectives that is the basis for handling any process improvement and operational implementation of organisational change successfully in even the smallest set up. It adds the perspective of the individual as a supplement to often referenced process improvement and change models, obtaining the ability of improving small and medium sized enterprises (SME's) with low costs, fast results and high rate of success.

Keywords

Software process improvement, organisational change, operational implementation, individual perspective, SME

1 Introduction

The Small and Medium sized Enterprises (SME's) we meet are often challenged by very limited overhead for long-term activities not directly linked to the business. Their perspective is to make the next project reach its goals and ensure continuous business.

Experience shows that many process improvement initiatives do not obtain the expected results, if any business improvement at all. It also shows that it is often very expensive and long-term before benefits start to come. To obtain a step in maturity from e.g. certified CMMI level 2 to level 3 takes at least 2 years for most organisations. At the same time the correlation between more and better businesses in the short run is low, when the organization e.g. follows the CMMI model default recommendations. The stories that more than half of all CMMI improvement initiatives fail, are also heard in the SME's.

Many investigations have tried to find the reasons for this low rate of success, though not been able to point to one single or few root-causes. What we have observed in some development organisation, is that a lot of effort is put into analysis of what changes to focus on, but very limited effort to ensure the new and better world is actually implemented to change the way individuals do their job. E.g. that an organisation of 100 developers can use more than EUR 100.000,- for an analysis of the best place to start, but then simply ask the middle management to do the implementation with very limited extra time and no extra budget.

We have also seen that organisations, in order to avoid putting development projects at risk, centralise the improvement. This result in a major effort for the centralised group is to obtain and keep resource allocation for the improvement tasks, while projects reduce the effort for implementation as they often do not see the benefit of being involved.

In short, seen from the small and medium sized organizations, the risk is high and probability of success is limited, e.g. for obtaining ROI or even reduction of internal problems. There will most likely be other, much more prosperous initiatives with less risk to invest the limited resources in…

This picture leads many organisations to only quick fix the problems on the short run. Problems do arise and need to be taken care of, but they see broad initiatives as high risk and very expensive. They reduce changes to a few people in one project, with no money for competent support or optimal solutions. Sometimes this does actually solve the concrete problem on the short run, sometimes not. In any case, they forget the optimisation when the team splits into new projects. The investment only gave a limited pay back.

In the investigation of the Danish company Systematic and how it obtained CMMI level 5, Pries-Heje, et al (2008), there is the finding that not before level 3 was obtained the organisation had learned how to do efficient process improvement that lasts.

In order to change the organisation we see a need to have a less detailed investigation and to invest more money in obtaining a lasting change.

We see a need for focusing on changing the things that are needed – here and now – and to implement the change in a way that reduces the project risk instead of increasing it.

We see a need for organisations to understand that doing process improvement requires structure, knowledge and skills in change implementation, to obtain success. Often this is not seen by the organization or not accepted to be a capability important enough to build up in a structured way.

We often meet companies with major process challenges, and to be able to help, we need an approach that companies quickly become comfortable with and where they feel *they* control the activities and expenses. It must be flexible to adapt to the situation in hand, taking the people and their view of the world into account. Most importantly, it must be able to predict and create visible improvements in both micro-steps and in larger initiatives, with a very high chance of success.

We believe an approach like this is beneficial for anybody, who is involved in operational process improvement!

2 Perspectives in Process Improvement

We see three main perspectives, which any organisation need to be aware of and use to define goals and set expectations from. This we base on the above and inspiration from CMMI/Bootstrap structure of processes and their dependencies doing process improvement, Agile development focus on creating value and risk minimization through focus on delivery with short intervals, Change Management as seen by Kotter (1996/2002) listing the foundation for success with change implementation and Weinberg (1997) focus on changes start with the individual and in the end rely on good change agents - and our own experience gained in working with SME's, saying you need to respect the short term focus and start where each company are to obtain success.

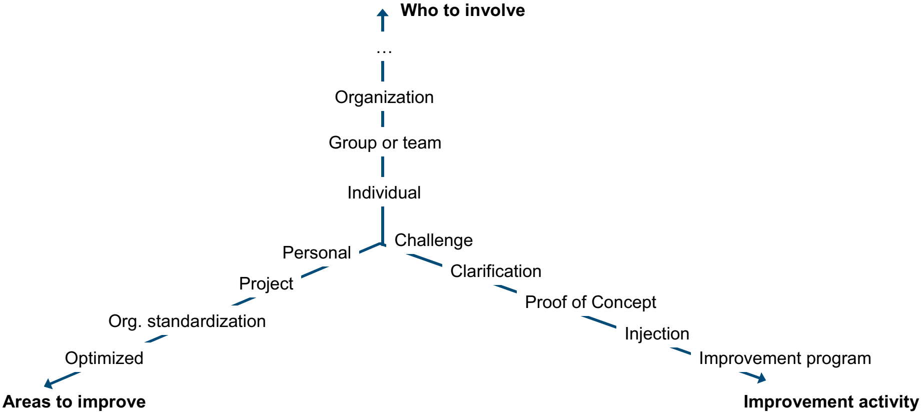


Figure 1 Framework for setting expectations and goals for Process Improvement activities

The framework is described in more details below.

An important point is that you do not need to do it all from scratch. The framework must be used to decide where to start and where to stop each iteration. You do one small step at a time, deciding for the next step, when you are ready to start it.

It is an important principle to finalize each step before moving resources to other tasks. This is an old truth e.g. described by Kurt Lewin in 1951. He called the principle Unfreeze-move-freeze and it is a very important point when changing people and organizations. If you stop before you have ensured the change has become a part of the organisation and people's daily work, people *will* go back to old habits and the way "we used to" and you wasted your change effort!

The ability to help is one more principle, not shown in the framework, but still very important! We see the words below by the Danish philosopher Søren Kierkegaard as a primary guide, when helping people and organisations. They were published in 1859 (here in a translation from 1998), but are still very true for anybody helping other people to improve:

"If One Is Truly to Succeed in Leading a person to a Specific Place, One must First and Foremost Take Care to Find Him Where He Is and Begin There.

This is the secret in the entire art of helping. Anyone who cannot do this is himself under a delusion if he thinks he is able to help someone else. In order truly to help someone else, I must understand more than he - but certainly first and foremost understand what he understands. If I do not do that, then my greater understanding does not help him at all.

If I nevertheless want to assert my greater understanding, then it is because I am vain or proud, then basically instead of benefiting him I really want to be admired by him. But all true helping begins with a humbling. The helper must first humble himself under the person he wants to help and thereby understand that to help is not to dominate but to serve, that to help is not to be the most dominating but the most patient, that to help is a willingness for the time being to put up with being in the wrong and not understanding what the other understands."

The learning to take from this is always to start with an understanding of people involved and the situation they see. Even though you challenge the way people work, always remember to respect their view. This is done from beginning to end in any improvement process, as it is not the change agent, but the people in the organisation that are expected to change – to work differently – to forget old habits and get new ones! To do that they do not need orders or cosmic truth! They need help, support and guidance. They need to see the benefit of changing themselves. They need to feel in control of their own future.

2.1 Who to involve

When starting to discuss process improvement you first need to identify the people and organisational elements to be involved. This is important in order to:

- Define a scope with clear boundaries making it possible to manage.
- Set expectations both for those who must be involved and for those that will not be involved.
- Take the first step in the sizing and set up of the activity. The more people to be involved, the more effort and risk will be in the activity before doing the freeze.

The defined levels are as follow:

1. **Individual:** The first level is a single individual or a couple of people. Discuss the improvements directly with the individuals that need a new way of working.
2. **Group or team:** Next level is to do processes in collaboration. Still, this will include only one manager or project manager. Handling the process improvement is with individuals present at one location, working closely together on a daily basis.
3. **Organisational:** An organisational entity with a more complex management structure, e.g. development departments with a matrix organisation with both a line and project manager, influencing the way people work. Now the handling of Stakeholders become more complex and need much more effort.
4. **Several organisations:** More than one organisational entity, e.g. distributed development on several locations. Consider this not just as a process improvement, but an improvement programme as it will be impossible to handle in one unfreeze-move-freeze operation.

2.2 The Area to Improve

The Second perspective is what to implement and to what level it needs to be implemented. It is to a certain extent the staged CMMI/bootstrap/Spice structure, though added the level of abstraction concerning organizational goals, starting from the individual and ending in managed organizational optimization. This is important in order to:

- Define a scope with clear boundaries within the processes that must be optimized and make it manageable.
- Set the expectations for capabilities of the people and organisation after finishing the unfreeze-move-freeze iteration.

- Identify and find competences and tools to be used for the implementation.

Processes are what people do. This is reflected here by each level of abstraction depending on the former.

1. **Personal:** The first and most concrete level for a process is the personal level, i.e. the extent to which the individual is capable of performing a given process optimally. Some processes are designed with focus on the individual, but any process has an individual level that needs to be in place, before the next level can work efficiently.
   Examples of personal processes are pair programming, performing a review for a peer, preparing documents, personal planning including estimation of tasks, quality control of own tasks, personal metrics and evaluation for optimization of the personal process.
2. **Project:** The next level is the processes that make a group or a team capable of working together. Examples could be Scrum, user stories, any project management activity also risk management, quality planning, monitoring and execution e.g. like test management. Also supporting processes like configuration management including change and defect management, effective reviews of different nature, measurement set up, … Note that this level can only be successful, if you also are capable of handling the first level of personal processes.
3. **Organisational standardisation:** When processes are used, the next level is to take them to become standardised in the organisation, making cross project and department collaboration and optimisation possible.
   Examples are to obtain agreement of commonalities and diversity in company processes and criteria for selection in each case. This is typically built around a management system and a quality department, though it can be done differently. It includes areas like managed training in processes and tools, structured maintenance and optimization of common processes or a common test platform for all projects.
4. **Optimized processes:** This refers to processes seen at CMMI level 4 and 5. As the author has not been working with these areas this will not be discussed further.

2.3 The Improvement Activity

The third perspective is extremely important, as it is about starting where people are! The idea is taken from Kotters (1996/2002) 8 step model, with the first step being to identify the reason for changing and the injection going through all steps in a short time span. Weinberg's (1997) focus on the individual is added, setting the optimal strategies and support for implementation.

The reason for discussing this is to:

- Define a scope with clear starting point and clear and measurable goals as ending point.
- Setting the expectations for the maturity of the implementation after the iteration is done and including the concrete activities.
- Offer an opportunity to ask all the questions, that enable you to learn what the persons you are about to help, really know and really need.

Even though you yourself as change agent, or just someone in the organisation to change, might know exactly what is needed, it is not the same as if everybody in the organisation is ready for it! We meet people at the following levels of change readiness and use the appropriate activities:

1. **Challenge:** Find out whether there is a potential benefit to gain. Is everything running smoothly? How does your group or organisation perform compared to comparable industries or in relation to a given maturity model? This can be done as a 2 hour informal meeting with a couple of key persons or be a set of structured sessions with many participants. Depending on your organisation, the situation and the goals you have.

2. **Clarification:** Find out where to start – where you will gain most. Again this can be short. In some cases, someone in the company often has a good idea of where to start. Though always do some checking and comparison with a maturity model like CMMI to ensure you do not start improving areas experienced to be high risk, if other areas are not in place. In addition it is recommended doing a session with key stakeholders, to start up the commitment for the changes to come. This is often sufficient for a 2-4 month long unfreeze-move-freeze iteration. If your company is looking for initiating a long-term strategic initiative, assessment or appraisals based on a normative model like CMMI, will be a good way to start. However, do remember to balance the effort for clarification with the effort and possibilities for changing your organisation.
3. **Proof of concept:** Will it work? Try it out without putting all projects at risk. This is like a pilot-project, with few involved persons to isolate parameters under test and focus on the process optimization and evaluation. You must be very aware that the goal is to demonstrate, how to succeed, not "just" run a pilot. In addition the involved persons must see the evaluation as open minded and that it may take any direction. The result is a decision to continue deployment, perhaps with some optimizations, or to close all further activities with the evaluated concept.
4. **Process injection:** Deploy! When taking the decision to implement a change, the challenge is to do the unfreeze-move-freeze as fast and smoothly as possible, as the transformation is often a chaotic, insecure and frustrating state for most people to be in. People will always be less efficient and it is a high-risk state for a project to be in. To obtain a fast transition and secure "freeze", a plan is created based on Kotter's (1996/2002) eight step model for implementing changes and Pries-Heje (2004) role model for the IT-diffusion process:
   - Reason for change is clearly defined.
   - Vision and goals stated with ending criteria.
   - Sponsors and stakeholders listed.
   - Those to do the implementation are listed with clear resource allocation, responsibilities and commitment.
   - Short term milestone including how to ensure results are visible to all stakeholders are defined.
   - How to ensure institutionalization is defined, including process measures.

   Use the principles of action-learning including support, coaching and short feedback loops, working with actual tasks during learning to increase empowerment, motivation, risk reduction and ensure project progress within hours or days. An injection has a complete cycle of 1-3 month. Not more!
5. **Improvement programme:** For large strategic initiatives more focus must be put on the involvement of senior management stakeholders. The unfreeze-move-freeze concept and management of the activities will need to be on more than one level of abstraction. The principles of the eight steps and roles to be manned are though the same.

2.4 Use the Framework for Maturing People and Organisation Step-by-step

The way to use the framework is straightforward.

Start with an open mind! Learn all you can about the involved persons' situation and view points. Why do these people need improvements? What business drivers can you find? Why do they feel a need to change? What are the primary challenges for doing development? What improvement would make a big difference for the business? Note that the answers must be said out loud in the group of stakeholders. If their view is not homogeneous, this is the time to find out!

If you find no reason for changing – recommend NOT to start an improvement initiative! You will create frustrations and not improvements! This we actually observe from time to time – that a group or a manager say they want to implement a new process, but when asked, they can not give any reason for doing it, except that it would be "nice to have!" That will not be sufficient for keeping priority! Don't do it!

Do this activity even though you might know the answers. The communication and common picture in the group of stakeholders is a basis for success with any improvement activity!

Then check that the people involved represent the scope of the improvements discussed and check if key stakeholders are missing. This can be at any level of the organisation. If so, get them involved or agree to do so very soon.

The above are the first steps in maturing the involved people to have a common understanding and agreement of the goals. Actually, this is the first step in a "Challenge".

Introduce the framework, one perspective at a time.

Using the business input you got in the beginning, pinpoint where the organisation and people are and discuss what focus would address the improvements needed, pointing to the three perspectives. Look for differences of opinion between the involved stakeholders and make sure these differences become visible.

Ensure that an agreed statement for why changes are needed is formulated and also an agreed statement of what bright future is expected to be obtained. Note this is not in 2 years time, but within the next 2-3 month!

Now you can easily point out one initiative to focus on, both an area to improve, the relevant group to involve and an activity to do! The stakeholders must agree to the goals and then you can formulate a concrete deliverable – a specific process that is improved, what improvement activity is involved and how to measure or indicate that the goal is obtained. In this way, expectations are clearly set and the tasks to do are clear for all involved.

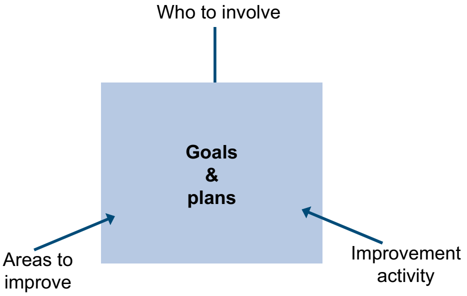


Figure 1 The framework gives input to plans and goals

3 *Examples*

In the following you will find two examples, selected to show that both agile methodology like Scrum and traditional process concepts like formal inspection, fit well into the framework and process injection.

3.1 Small Software Development House Increase in Size

A small software development company of 8 software developers were originally started based on a completely new type of product. Most work was done as prototyping together with the customers. But lately their role has become more like a traditional supplier, as the customers also mature within the business area.

They knew changes needed to happen, as they could see it became still more difficult for them to deliver as agreed. No challenge needed! The question of where to start and what to do, where unknown to the company. They needed a clarification.

A clarification session was therefore the first step. Managers, technical sales and developers were placed in a room and taken through a 2 hour managed process to identify and prioritise the most important areas to improve and the most important areas to ensure continued.

The clarification results showed that the informal and flexible handling of daily work should continue, but also that visibility of progress and ability to plan and to have a high chance of delivering on time was needed, in parallel to handling small tasks from customers expecting results in weeks and not months.

We ended up agreeing Scrum would be the solution that fitted their needs, changing everything concerning development in the entire company. We planned an injection.

First iteration was to train the personnel and implement Scrum in Development, including getting the Product Backlog and its update every month in place – and to ensure NOTHING was done, unless it is prioritized in the backlog! This was in place in one Sprint taking one month.

Second iteration was to mature the process and anchor ownership and process responsibility by maturing the ScrumMaster and implement Scrum in the quality system.

After two months a mature Scrum implementation was running! The result is that now release is done on time and when relevant key customers are even invited to see and discuss intermediate results at the review meeting once a month.

3.2 Medium Sized Organization are Challenged by Quality

In a development department of approx. 60 developers doing products with software, hardware and mechanics, they where challenged by too many quality issues showing up during initiation of production for new products.

The organisation had done an analysis of root-causes and found that problems did not originate from a single source or process step, but originated from most activities and areas in development. Several initiatives where launched and additionally they had a feeling that formal review would also benefit, but how?

When we were involved, the organisation therefore did not need a challenge. They knew improvements were needed. They were also clear about the next step to take. It was checked and found that there was a broad support for the selected improvement: An injection of formal review. The task was therefore to tailor Fagan Inspection to their specific needs and do a detailed plan for implementation. The management and the task force for the initiative accepted this. A few days later the injection was running with training, coaching, support, set up of process measurement and implementation in the quality system. An evaluation done 3 month later showed that defect levels improved and that reviews where done with positive feedback. There were still opportunities for using reviews in even more areas and for more deliverables. A process responsible was trained and placed in the organisation, making the organisation capable of doing the next iteration themselves.

4 Conclusion and Discussion

Our experience in working with process improvement show that mainly two things go wrong for SME's:

- The perspective of the individual is often forgotten
- Managers and change agents are often abstract in their communication of outcome, business benefit and consequences.

This results in lack of help with the improvements from those to perform and those to sponsor the improvements and they end up being "resistance to change".

We have used the described framework both as a tool for communication with management and for planning/follow-up on improvement activities. We often get the response that "…now it makes much more sense!" Often when the dialog starts, people find it difficult to discuss process improvement, but when the framework is presented and we are specific, discussing the challenges which the company experiences and how it fits into the framework, the enthusiasm increases. Now they can see which specific actions can improve their situation, the amount of resources this will require and the fact that this will be manageable with a high chance of success.

The framework is very concrete in conveying the meaning and gains for the individual, including what each person needs to do to meet the goals. The time of "personal chaos", when moving from old habits to new ones, is extremely short. That reduces risks and frustrations. E.g. communicating an injection is to use a calendar, show when the one hour training is done and when they will be doing a session with a well-defined deliverable - mentioning this is done with support available on site. And so on… This may give a few questions, but we have only met resistance when the up front challenge was not done well, thereby not resulting in clear reasons for change and visions for where to end. Visions that everybody could agree to!

The plan made will be very concrete and often goals are measurable or at least it is clear which criteria mean success or not! When defining the goals, we sometimes see that it is not understood why this is important to do in this details, but half way through, when the goals have been used several times to get the improvement back on track, this is understood!

We have though seen that larger companies have difficulties to see e.g. an improvement organisation already in place, fit into the framework. We believe this happens because the framework does not itself visualise how to organise improvement programs. It has never been the intension to model this, but to focus on the concrete implementations, being part of such a programme. Therefore this needs to be added, when discussing larger improvement set ups.

We are sure that following this framework and approach will increase the chance of success in many improvement activities, as even the largest process improvement initiatives in the end is about changing individuals!

If you do not agree, we hope you will use the time to challenge us…

Literature

Gerald M. Weinberg: Quality Software Management, Vol. 4, Anticipating Change,
1997, Dorset House Publishing

Jan Pries-Heje: Role models for the IT diffusion process,
2004, Kluwer Academic Publishers

Jan Pries-Heje, Jacob Nørbjerg Ivan Aaen and Thomas Elisberg: The road to high maturity,
2008, Software Innovation Publisher

John P. Kotter: Leading change,
1996, Harvard Business School Press

John P. Kotter, Dan S. Cohen: The heart of change,
2002, Harvard Business School Press

Kurt Lewin: Field Theory in Social Science.
1951, New York: Harper & Row

Søren Kierkegaard: ”The Point of view for my work as an author”,
1998, Princeton University Press, chapter IA, §2, p. 45. (First published in Danish in 1859)

5 Author CVs

Kurt S. Frederichsen, Center for Software Innovation

Kurt S. Frederichsen is Process Improvement Specialist at Center for Software Innovation. He received a B.Sc. E.E. from Engineering College of Aarhus, Denmark.
In 1995 he took upon him to start a quality and process improvement function in Danfoss Drives software development. This expanded to support a development organisation distributed in three countries and to support both development of software, hardware and mechanics. Afterwards he held a position of process responsible for a strategic CMMI improvement initiative in Maersk Data Defence. In his present position as Process Improvement Specialist he assists SME's in reducing risks and optimising product development. He designed the method described in this paper for this specific purpose.

Center for Software Innovation is a knowledge centre specialised in creating growth in SME's through innovative use of hardware and software technology and is partly funded by the public to proactively search and approach companies with growth potential. The centre do this by focusing on supporting companies in adding intelligence to products, raising process capabilities and enhancing use of market opportunities.

Managing the Software Process with a Software Process Improvement Tool in a Small Enterprise

García, I.[1]; Pacheco, C. [1]; Calvo-Manzano J. A.[1], Cuevas, G.[1], Mendoza, E. [2]; San Feliu, T. [1]
[1]Languages and Information System and Software Engineering Department
Faculty of Computer Science, Polytechnical University of Madrid
{igarcia, cpacheco@zipi.fi.upm.es}, {joseantonio.calvomanzano, gonzalo.cuevas, tomas.sanfeliu}@upm.es
[2]Postgraduate Department
Technologocial University of the Mixtec Region. Mexico
eloy@mixtli.utm.mx

Abstract

Top-down process improvement approaches provide a high-level model of what the process of a software development organization should be. Such models are based on the consensus of a designated working group on how software should be developed or maintained. They are very useful in that they provide general guidelines on where to start improving, and in which order, to people who do not know. However, the majority of them have only worked in scenarios within large companies. We aim to help small software development organizations to adopt an iterative approach by providing a Process Improvement Web-based Tool. This paper presents research into the proposal that a small organization may use this tool to define and implement a set of agile project management practices that can be strengthened using the CMMI-DEV 1.2 model as reference.

Keywords

Software process improvement, software tools, assessment, questionnaire, improvement plan.

1 Introduction

Nowadays, software processing in an industrial environment is key to the productivity and efficiency of development activities and competitive strategies against adversaries. However, since the beginning of the 1980s the software industry has tried to increase its quality and productivity by applying new methods and techniques. It has been recognized that unfortunately the fundamental problem for many companies is the incapacity to manage the software process. In recent years there has been on-going demand for better services and functionality in software products. Many methods, techniques and tools have been developed. Nevertheless, software products still suffer from excessive costs, delays in delivery and low quality. To satisfy the various requirements for software processing both large companies and Small Enterprise (SEs) have made a central effort in Software Process Improvement (SPI). As a result, SPI has emerged as a new way of solving these problems. An indicator of this is the increasing number of international initiatives related to SPI, such as Capability Maturity Model Integration for Development v1.2 (CMMI-DEV) [29], ISO/IEC 15504:2004 [14], SPICE [13], IDEAL [17], and ISO/IEC 12207:2004 [12]. These SPI models have been presented as an alternative to achieve an increase in the quality of services and products that software organization provides.

This interest in software improvement in large enterprises is now being extended to small settings. However, the problem is the high implementation cost, which is independent of the size of the company [19]. Because models have been developed for large enterprises, only a few small and medium software organizations are aware of them. It is recognized that for small settings, projects or teams, process improvement efforts are difficult and frequently overwhelming; however, the need for improvement remains and cannot be ignored. Nevertheless, SPI is not explored sufficiently by IT departments in many organizations; hence any development of an SPI initiative becomes completely chaotic as important concepts are unknown or not applied. Apparently, SEs typical issues including the software process management activities are as follows; no principles for the manner of performing tasks including plans and procedures, ad-hoc task performance relying only upon the experience of practitioners, and an insufficient number of staff with appropriate skills. Furthermore, small software development organization needs an automated tool supporting the way it effectively manages its software processes [31].

2 Related Work

We have researched the characteristics of typical SPI Tools for small software development organizations and projects. Unfortunately, there is not too much research about SPI Tools development to help the improvement initiatives in SEs. SPI Tools generally automate the process of assessment based on the standard model framework for software process, such as ISO/IEC 15504:2004 and CMMI [28]. The existing SPI Tools usually include some or all of the following characteristics:

- Collecting and managing general company information, assessment goals, project and process instance information, assessment items, etc.
- Selecting which processes to assess.
- Generating questionnaires.
- Modifying the software process model for domain adaptation and the changed versions of the process model.
- Self-evaluating tasks by software practitioners.

However, the main objective of these tools is to conduct the process assessment to ensure a project's compliance with specific standards in software development organization. Thereby they can be intended only to automate the process of software process assessment. All software tools for implementing and maintaining improvement initiatives can be divided according to the model under maintenance: CMMI or ISO/IEC 15504. The other aspect of division is open source software (including academic tools) and commercial software. Table 1 summarizes the current SPI tools from process model and assessment method perspective and provides a brief description of each one.

Table 1. Academic Tools for SPI.

Name	Description
SEAL OQ [9]	SEAL of Quality SPICE assessment tool is an automated tool to support the conduct of assessments in Phase 2 of the trialing period using the embedded model described in Part 5 of ISO/IEC 15504.
A visual Approach to Software Process Improvement [10]	The tool software used MS Access and its aim was to explore software process visualization.
PIASS [22]	Process-Improvement Activity Support System was developed as an integrated environment that provided tools necessary to make an assessment of targeted software processes.
SPIS [7]	The Software Process Improvement support System indicated 'how to' perform SPI activities.
Process Management System [5]	The system provides two different approaches to its use. One was an internal assessment and improvement process at the level of software process appraisal. The other was a self-controlled improvement process.
Taba Workstation [20]	The Taba Workstation is a Process-centered Software Engineering Environment (PSEE) that supports software process definition, deployment and enactment.
KMT [1]	Knowledge Management Tool facilitates the SPI implementation.
Knowledge-Based System [2]	The Knowledge-Based System evaluates an organization at a determined CMM level and as such limits the need for the services of an auditor in those cases in which the system's response complies with the requested CMM level and the necessary associated skills.

In analyzing commercial software most attention was paid to the commercial tools of one of the leading companies HM&S; these tools are designed for working both with ISO/IEC 15504 and with CMMI. Table 2 summarizes a list of commercial tools that provide guidance in SPI initiatives.

Table 2. Commercial Tools for SPI.

Name	Description
CMMI v1.2 Browser [33]	CMMI v1.2 Browser allows users to navigate easily through the CMMI-DEV. This CMMI browser covers the continuous as well as the staged representation.
CMM-Quest [25]	This software tool rates and analyzes the software development processes compliant to CMMI-DEV v1.2 and uses ISO/IEC 15504 as its assessment method.
Appraisal Wizard [3]	Appraisal Wizard is based on the SCAMPI [18] method and supports almost all CMM models. This tool assigns all values to different assessment instances (practices, objectives, process areas) and provides large and detailed support.
SPICE 1-2-1 [30]	SPICE 1-2-1 assigns values to base and generic practices; it has a medium rate of difficulty when organizations try to use it. Also, SynEval (another tool from SynSpace) helps the Assessor and/or particularly the Lead Assessor to argue around the results from several assessments to analyze them.
IME Toolkit [11]	Management Information Systems (MIS) developed the free tool called Interim Maturity Evaluation Toolkit based on CMMI models. IME Toolkit assigns numerical values to practices and generates a score for process areas.

MKS Integrity Suite [21]	The MKS Tool fully supports SPI initiatives and complies with certifications such as CMM and CMMI. The MKS Tool provides automated audit tracking and control.

However, such commercial software is defined by a relatively large price tag (from $900 to $15,000 USD) which is out of the price range of SEs. Its main features are a well organized subsystem of the assessment presentation, a sufficient amount of tools for maintaining the process improvement, and integration with other tools. However, despite all the advantages affirming the existence of efficient assessment and maintenance tools nowadays, there are also some considerable drawbacks. Furthermore, few tools can really help a SE, taking into account its particular circumstances; but hardly any provide an automated mechanism to create improvement plans. That is, there is a limit for being able to provide more helpful and diverse support for establishing SPI initiatives and managing the whole process of SPI.

3 An Automated Tool to Support SPI Initiatives in SEs

According to Jones [15], three major investment elements are involved in CMMI-based improvement: *assessment*, *definition/infrastructure support*, and *deployment*.

However, our experience shows that small organizations have a disadvantage with resources for assessment and definition but have a distinct advantage in deployment. A frequent misconception about implementing CMMI and IDEAL is that they work only for large organizations (their cost and complexity appear to make it impractical for small businesses to implement). Nevertheless, there is a lot of researches that contradict this assumption [32] [24] [23] [26]. We think that this idea may be supported if we are able to design and implement a Web-based tool to manage the improvement information of any small organization and guide it to improve its software process. To avoid this issue, the CBA-IPI [6] has been used to self diagnose and guide the SPI implementation efforts. Since then, some gap analysis and internal evaluations using SCAMPI have been performed in order to guide the SPI activities. We implemented a Questionnaire-Based Appraisal (QBA) as an assessment tool because it can be applied to many people, it is cost effective and non-invasive, provides quantitative data, and the results can be analyzed promptly [16]..

3.1 Components of the tool

We are trying to provide a Web-based tool (called SysProVal) that establishes an iterative approach to Process Improvement (PI) which a small organization could adopt. Following this approach to PI, the focus of the first step would be to understand what exists in the organization and determine what causes significant problems. Then solutions could be devised in the action plan and evaluated in pilot studies or even controlled experiments. Only after a solution is found to be effective and efficient, it should be integrated into the existing process or that process be modified. Such an approach is inspired by the Quality Improvement Paradigm (QIP) recommended by several researchers and practitioners [4]. SysProVal consists of an SPI manager and mechanisms for: comparing the current practices with CMMI-DEV-adapted practices for SEs; assessing the selected process; generating an adequate improvement plan; and using the iterative elements of the tool. SysProVal provides three types of user interface components: (1) the top management interface, (2) the project manager interface, and (3) the SysProVal interface.

3.2 The Top Management Interface

The purpose of the top management interface is to support the whole range of SPI activities in an integral manner in terms of necessary guidance for SPI activities, communication support, and learning support. The interface provides access to:

1. The *Process selection browser,* which provides a checklist indicating to the top management what processes can be selected. Each process selected is stored in the *Organizational process repository* and is associated with the description stored in the *Effective practices for SEs component*.

2. The *Role selection browser,* which allows the top management to select who is the project manager to be evaluated. They can also decide whether to delegate the internal role to the SysProVal manager, or address the SPI initiative with an external assessor.

3. The *Communication interface,* which allows the top management to send email messages using links incorporating objects from other interfaces, such as reports, questionnaires, or new assignments. Through this interface, top management can be informed about the progress of the SPI initiative, consulting results, reports, or progress status.

4. The *Reports generator,* which allows the top management to obtain any information on the SPI cycle at any time; they can control the performance of its project managers through the assessment process and obtain the final results and graphics derived from the entire process.

3.3 The Project Manager Interface

The purpose of the project manager interface is to obtain all information about organizational practices to develop software products and to allow them to have a good understanding of what is going on in their development process (weaknesses and strengths).

1. The *Tutorial interface* helps the project manager to obtain all knowledge needed to use the tool, interpret the results, understand each question, and improve its work.

2. The *Visualization interface* provides the project manager with the tools to draw a new process or reuse existing ones (in accordance with CMMI-DEV activities).

3. The *Current status report interface* presents a categorized level of performance, in accordance with the assessed process. The project manager can meet its own level of performance. This interface just provides performance level results to project managers; the entire results of the organization can only be reviewed by high-level management through the reports generator.

3.4 The SysProVal Interface

The purpose of the SysProVal interface is to allow the top management of an SE to explicitly state the goals of the SPI activities, and to have a good grasp of the current progress of SPI activities in all improvement projects. Moreover, project managers can directly participate in the SPI activities through assessments and Improvement Plan (IP) generation.

1. The *Questionnaires interface* guides project managers to obtain all knowledge from their daily labor. Previously, top management had selected which process to be evaluated or what process to be modified. This interface uses the '*levels of performance*' from the two-phase mechanism proposed in [8].

2. The *IP interface* generates the improvement plans for each improvement project conducted. This interface uses knowledge management using databases to manage both information and people.

4 Implementation and Evaluation

4.1 Implementation

We have developed the proposed system in PHP using MySQL as a database management system. PHP is free software released under the PHP License. We think that these conditions are favourable and affordable for any SEs. This work is structured according to the first three phases of IDEAL. Although these phases are presented sequentially, in most cases they are overlapped.

4.1.1 Initiating

The SPI initiating phase was clearly covered and came from the executives of the organizations; therefore, it was not necessary to get any buy-in or work to get executive support and commitment. According to Mondragon the PI infrastructure has three groups [19]: the management steering group (MSG), the engineering process group (EPG) and, the technical working group (TWG). For small settings, IDEAL recommends at least one full-time employee. Satisfying the one full-time employee requirement recommended in very small organizations is a challenge. The first question posed was from which part of the organization should the PI leader be drawn? Should the PI leader be a top executive, a manager, or a practitioner? We implemented the third approach (a practitioner as a leader of the PI effort) and it is currently providing the best results. The second important issue in the initiating phase is the training concept. We developed an SPI tutor that satisfies the knowledge acquisition process according to the SEs necessities. This is not a certification course or even a formal course. This tutorial manages the lessons and promotes the inclusion of new topics as the improvement project progresses. If the performance of the project manager is low during the assessment, the tutorial recommends topics and alerts top management for monitoring and control.

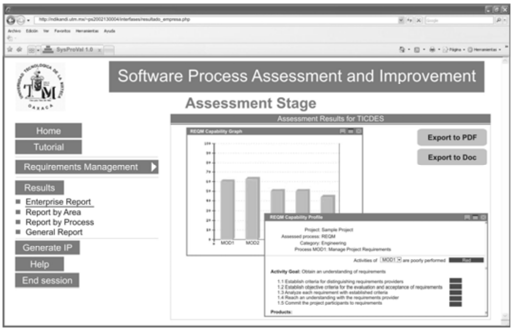
4.1.2 Diagnosing

In order to characterize the initial state of the organization's software processes as well as to assess the improvements done through the SPI cycle, we implement an assessment mechanism [8] inside SysProVal as a *"seek step"* of PI efforts. This mechanism uses closed questions and limits the number of possible answers to seven. Each type of answer has a unique interpretation and indicates the *performance level* of a practice. The level-perform-answers determine the percentage in which each practice is performed. This varies from 'Never' with a value equal to 0, 'Rarely if ever' with a value equal to 1, 'Sometimes' with a value equal to 2, 'Usually' with a value equal to 3, and 'Always' with a value equal to 4. The validity-answers don't have numerical value. Giving a specific weight to each answer will enable SysProVal to easily analyze the results of the assessment and to identify which practices are common within the whole organization and which ones are not performed at all. The assessment mechanism proposed here is part of the *questionnaires interface* and has been based on the two types of practices established by the CMMI-DEV dividing it into two phases. The first phase is related to specific practices while the second phase is related to generic practices. As an example, the following figure shows the results of the initial SysProVal assessment for one small organization through the *current status report interface*. The charts showed in Figure 1 summarize the percentage of *effective practices for SEs component* already addressed in the existing processes of the organization, and the results are automatically grouped by SysProval according to CMMI-DEV. This information is sent to top management by the *reports generator*.

4.1.3 Establishing

In the *"improvement step"*, SysProVal sets the priorities, then develops the approach and finally generates the action plan to improve the current process with the obtained data from the previous phase. The outcome of the *"improvement step"* is a detailed implementation plan that is composed of eleven domains: specific actions, schedules, milestones, deliverables, decision points, resources, responsibilities, measurements, tracking mechanisms, and risk management and mitigation strategies (see Figure 2).

Clearly, the *IP interface* of the SysProVal tool, when applied to software projects in small organizations, produces a detailed plan with very similar features to those of large organizations. Therefore, given the organizational experience of using CMMI models, it is natural to think about the CMMI-DEV as the supporting process for planning, tracking, and controlling the SPI implementation activities for the improvement team. The IP generator organized the obtained information from *“seek step”* using databases. The most primary and preliminary requirement for a knowledge management in SysProVal is to manage the acquired data from the *assessment interface*.



Project: Sample Project
Assessed process: REQM
Category: Engineering
Process MOD1: Manage Project Requirements

Activities of MOD1 are poorly performed

Activity Goal: Obtain an understanding of requirements

1.1 Establish criteria for distinguishing requirements providers
1.2 Establish objective criteria for the evaluation and acceptance of requirements
1.3 Analyze each requirement with established criteria
1.4 Reach an understanding with the requirements provider
1.5 Commit the project participants to requirements

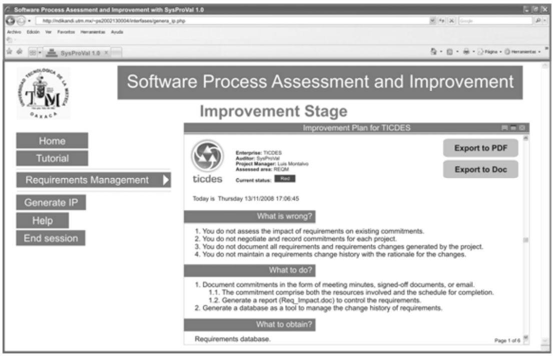
Figure 1. **An screenshot of reports generator**

The contents are obtained through the recording of events and experiences in a specified format and the capturing of good and bad practices from the web-based assessment. Conversion of performance levels in the form of information and storing them in the database is achieved through the engineering technique of data-mining. The transformation identifies important aspects of the knowledge domain, for example, notions, attributes, activities and associations with other process areas. SysProVal identifies similar notions and these are clustered based on their attributes (*poorly performed*, *partially performed*, and *adequately performed*). Giving a raw text to DMS, the captured activities are represented in the form of eleven knowledge domains and are stored in an object-relational data model [49] of the organization’s processes.

4.2 Evaluation

The main objective of this research is to demonstrate that a small organization may use SysProVal and the CMMI-DEV model as a framework to strengthen defined agile project management practices, improve the project performance, and achieve high CMMI-DEV capability levels. We think that this idea may be applicable to other process areas, but the data that has been gathered so far corresponds to project management activities, thus we decided to limit the current scope research to the project management category of the CMMI-DEV model.

Our first goal is to articulate capability level 2 compliance with the defined Web-tool discussed in this paper. This work is being carried out, and current findings will be presented here. Next we will articulate capability levels 3 and 4. Theoretical Capability Level 2 and Level 3 compliance will be documented by third quarter 2009, and actual Capability Level 2 Compliance is expected to be confirmed during an official appraisal planned for one of the organizations in the first quarter of 2009.



Software Process Asessment and Improvement with SysProVal 1.0
http://ndikandi.utm.mx/~ps2002130004/interfases/genera_ip.php
Archivo Edición Ver Favoritos Herramientas Ayuda
SysProVal 1.0
Página Herramientas
Home
Tutorial
Requirements Management
Generate IP
Help
End session
Export to PDF
Export to Doc
ticdes
Page 1 of 6

Software Process Assessment and Improvement

Improvement Stage

Improvement Plan for TICDES

Enterprise: TICDES
Auditor: SysProVal
Project Manager: Luis Montalvo
Assessed area: REQM
Current status: Red

Today is Thursday 13/11/2008 17:06:45

What is wrong?

1. You do not assess the impact of requirements on existing commitments.
2. You do not negotiate and record commitments for each project.
3. You do not document all requirements and requirements changes generated by the project.
4. You do not maintain a requirements change history with the rationale for the changes.

What to do?

1. Document commitments in the form of meeting minutes, signed-off documents, or email.
    1.1. The commitment comprise both the resources involved and the schedule for completion.
    1.2. Generate a report (Req_Impact.doc) to control the requirements.
2. Generate a database as a tool to manage the change history of requirements.

What to obtain?

Requirements database.

Figure 2. An screenshot of reports generator

Project Planning (PP) and *Project Monitoring and Control (PMC)* were already analyzed for Capability Level 2 Compliance. We will continue to gather data for the implementations of these process areas and we will articulate *Risk Management (RSKM)* and then *Integrated Project Management (IPM)*, *Organizational Process Definition (OPD)* and *Organizational Process Focus (OPF)* development to prepare for Capability Level 3. As a direct result of this project, information regarding to the small settings community and the issues associated with software development were generated. This information, however, needed to be extracted, organized, and analyzed with SysProVal, in order to be useful in addressing the above research questions. The data consists of artifacts generated as a result of the SysProVal evaluation.

The CMMI-DEV practices provide the environment and structure for this research. As part of this study these researchers created an advisory group for a small organization to provide guidance for the project and insight into the environment of this organization.

5 Lessons Learned and Preliminary Findings

We have reported on our early experiences in using a Web-based tool as a foundation for implementing the CMMI-DEV in small organizations. Our research has shown that SysProVal makes the CMMI-DEV implementation easier through the first three phases of the IDEAL model. SysProVal has good coverage of these phases at a project level. However, it is important to ensure that the action plan (derived from CMMI-DEV) is followed faithfully by all the teams and project managers in the organization. To this end, the *"seek step"* mechanism proved to be a convenient and useful diagnostic tool. As mentioned above, SysProVal has good coverage of the *"seek and improve"* steps at team level. Therefore, most of the CMMI-DEV implementation has to be at organizational level. Even if all the teams are following the CMMI-DEV by the book, there are still many organizational aspects that have to be covered. Implementing an initiative to provide an SPI tool oriented to small settings and focused on the SEI family of products was a good decision. This family of products includes models, an implementation lifecycle model (IDEAL), assessment and appraisal methods. These products provide a consistent and complementary set of tools that facilitate the implementation of an SPI initiative. However, it is too difficult to adopt them in very small settings without a specialized guide. SysProVal is a huge shortcut to implementing such models. A SysProVal launch was performed to define the detailed plan for the establishing phase. Table 3 shows a summary of the number of major activities and estimated effort for the implementation of each level related to CMMI-DEV (Levels 2 and 3) in the assessed organization, according to the plan provided in the SysProVal tool. The row called Set Infrastructure refers to the set of activities required as prerequisites before starting the implementation of all activities related to CMMI-DEV levels. SysProVal covers the first three phases of the IDEAL model to achieve a PI in software organizations through adaptation of CMMI-DEV in small settings. Actually, we are validating the Acting and Learning phases.

Table 3. Commercial Tools for SPI.

	Number of major Activities	Estimated Time (Hours)
Set Infrastructure	12	320
Level 2	60	675
Level 3	65	1300

6 Conclusions

Though CMMI and ISO/IEC 15504 have exploded onto the market as models to follow when organizations want to apply process improvements, there are many organizations that are still not using these models. The CMMI is considered to be one of the best known models that focus on software process improvement for achieving quality software in small settings. The CMMI-DEV, however, is relatively new, so there has been little research written on which data collection and action plan generation tools can be employed when using the CMMI-DEV approach under the special conditions of small organizations. This research, therefore, developed an instrument to evaluate the current status of Project Management practices. Its purpose was to investigate its feasibility in small settings and to influence the direction of future research. One limitation of this study is the generalization of its findings based on the limited amount of data collected and analyzed relative to the number of small organizations. This suggests that this qualitative study will be increased by quantitative studies to strengthen the data supporting the need and applicability of SysProval to the small organization community.

Acknowledgement

This paper is sponsored by ENDESA and Everis Consulting Foundation companies through “Research Group of Software Process Improvement in Latin America”.

7 References

1. Alagarsamy, K., Justus, S. & Iyakutti, K. “On the Implementation of a Knowledge Management Tool for SPI” *Proc. of the International Conference on Computational Intelligence and Multimedia Applications*, ICCIMA 07. IEEE Computer Society, pp. 48-55. 2007.
2. Andrade, J., Ares, J., Garcia, R., Rodriguez, S., Seona, M. & Suarez, S. *“A Knowledge-based System for CMM Evaluation”* Studies in Computational Intelligence. Springer Berlin / Heidelberg, 134: 183-191, 2008.
3. Appraisal Wizard, Formal or Informal Appraisal Tool, 2009. Available on: http://www.isd-inc.com
4. Basili, V., Shull, F. & Lanubile, F. “Using Experiments to Build a Body of Knowledge” *IEEE Transactions on Software Engineering*, 25(4): 456-474, 1998.
5. Choi, Y., Lee, ES. & Ha, S. “The Management of Software Processes with Software Process Improvement Tool based on ISO 15504” *Proc. of the 7th International Conference on Advanced Communication Technology*, ICACT 2005. IEEE Computer Society, pp. 933-936. 2005.
6. Dunaway, D.K., Masters, S. *CMM-Based Appraisal for Internal Process Improvement (CBA IPI). Method Description*. Technical Report CMU/SEI-96-TR-007. Carnegie Mellon University, Software Engineering Institute, Pittsburgh. 1996.
7. Fukuyama, S., Miyamura, S., Takagi, H. & Tanaka, R. “Validating the ISO/IEC 15504 Measure of Software Requirements Analysis Process Capability”, *IEICE TRANSACTIONS on Information and Systems*, E83-D(4): 747-756, 2000.
8. Garcia, I., Calvo-Manzano, J., Cuevas, G. and San Feliu, T. “Determining Practice Achievement in Project Management Using a Two-Phase Questionnaire on Small and Medium Enterprises” *Proc. of the European Systems and Software Process Improvement and Innovation Conference*, EUROSPI 2007, LNCS 4764, pp. 46-58, Springer-Verlag Berlin Heidelberg. 2007.
9. Him Lok, R. & Walker, A. J. “Automated Tool Support for an Emerging International Software Process Assessment Standard” *Proc. of the Third International Software Engineering Standards Symposium; Emerging International Standard,* ISESS 1997. IEEE Computer Society, pp. 25-35. 1997.
10. Hunter, R. & McCallum, C. “A visual Approach to Software Process Improvement” *Proc. of the Software Process Improvement Conference*, SPI’98, 1998.
11. IME Toolkit, 2009. Available on: http://www.man-info-systems.com/index_files/FreeTools.htm

12. ISO/IEC 12207:2002/FDAM 2: Information Technology – Software Life Cycle Processes. International Organization for Standardization: Geneva. 2004.
13. ISO/IEC TR 15504:1998(E): Information Technology – Software Process Assessments. Parts 1-9. International Organization for Standardization: Geneva. 1998.
14. ISO/IEC 15504-2:2003/Cor.1:2004(E): Information Technology – Process Assessment – Part 2: Performing an Assessment. International Organization for Standardization: Geneva. 2004.
15. Jones, J. "Process Improvement in a Small Company". *Proc. of the First International Research Workshop for Process Improvement in Small Settings*, CMU/SEI-2006-SR-001, pp. 251-265. 2006.
16. Marciniak, J., Sadauskas, T. "Use of Questionnaire-Based Appraisals in Process Improvement Programs" *Proc. of the Second Annual Conference on the Acquisition of Software-Intensive Systems*. Carnegie-Mellon University. Software Engineering Institute. Arlington, Virginia. USA, 2003.
17. McFeeley, B. *"IDEAL: A User's Guide for Software Process Improvement"* CMU/SEI-96-HB-001, Software Engineering Institute, Carnegie Mellon University. 1996.
18. Members of the Assessment Method Integrated Team. *Standard CMMI® Appraisal Method for Process Improvement (SCAMPI), Version 1.1*, CMU/SEI-2001-HB-001. Software Engineering Institute, Carnegie Mellon University. Pittsburgh, PA. 2006.
19. Mondragón, O. "Addressing Infrastructure Issues in Very Small Settings" *Proc. of the First International Research Workshop for Process Improvement in Small Settings*, CMU/SEI-2006-SR-001, pp. 5-10. 2006.
20. Montoni, M., Santos, G., Rocha, A., Figueiredo, S., Cabral, R., Barcellos, R., Barreto, A., Soares, A., Cerdeiral, C. & Lupo, P. "Taba Workstation: Supporting Software Process Deployment based on CMMI and MR-MPS.BR" *Proc. of the 7th International Conference on Product Focused Software Process Improvement*, PROFES 06, LNCS 4034, pp. 249-262. 2006.
21. MKS Tool, 2009. Available on: http://www.mks.com/process_improvement
22. Nakakoji, K. *"PIASS: Process-Improvement Activity Support System"* Technical Report, SRA-SEL-97081, Software Engineering Lab., SRA Inc., Tokyo, Japan, 1997.
23. Oktaba, H., Garcia, F., Piattini, M., Ruiz, F., Pino, F. & Alquicira, C. "Software Process Improvement: The Competisoft Project" *Computer*, 40(10): 21-28. 2007.
24. Santos, G., Montoni, M., Vasconcellos, J., Figueiredo, S., Cabral, R., Cerdeiral, C., Katsurayama, A., Lupo, P., Zanetti, D. & Rocha, A. "Implementing Software Process Improvement Initiatives in Small and Medium-Size Enterprises in Brazil" *Proc. of the 6th International Conference on the Quality of Information and Communications Technology*, QUATIC 2007. IEEE Computer Society, pp. 187-198. 2007.
25. Self Assessment Tool CMM-Quest, 2009. Available on: http://www.cmm-quest.com/
26. Shen, B. & Ruan, T. "A Case Study of Software Process Improvement in a Chinese Small Company" *Proc. of the International Conference on Computer Science and Software Engineering*, ICCSSE 08. IEEE Computer Society, pp. 609-612. 2008.
27. Syazwan, M., Benest, I., Evans, A. 6 Kimble, C. "Knowledge Modeling for developing knowledge management systems" *Proc. of the 3rd European Conference on Knowledge Management*. Dublin, Ireland. pp. 15-25. 2002.
28. Software Engineering Institute. *CMMI for Systems Engineering, Software Engineering, Integrated Product and Process Development, and Supplier Sourcing (CMMI-SE/SW/IPPD/SS, V1.1). Continuous Representation.* CMU/SEI-2002-TR-011, Software Engineering Institute, Carnegie Mellon University. 2002.
29. Software Engineering Institute. *CMMI for Development (CMMI-DEV, V1.2).* CMU/SEI-2006 TR-008, Software Engineering Institute, Carnegie Mellon University. 2006.
30. SPICE 1-2-1, 2009. Available on: http://www.synspace.com/tools.html
31. Young, H., Fang, T. & Hu, C. "A Successful Practice of Applying Software Tools to CMMI Process Improvement" *Journal of Software Engineering Studies*, 1(2): 78-95, December 2006.
32. Wangenheim, C. G. v., 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.
33. Wibas CMMI Browser, 2009. Available on: http://www.cmmi.de/cmmi_v1.2/browser.html#hs:null

8 Author CVs

Garcia Ivan

He has the PhD studies in Software Engineering at Polytechnic University of Madrid and is full-time professor at the Postgraduate Division of Technological University of the Mixtec Region, Mexico. He is author of several international papers related to Software Engineering and, more specifically, Software Process Improvement. Garcia is member of the Technical Committee of the International Association of Science and Technology for Development (IASTED) and also is a reviewer of a number of international journals and conferences in Mexico. He has participated in more than 15 projects related to software project management and software process improvement in the Mexican software industry. In Latin America, Garcia has participated in several institutional research projects in industry (IBM Global Services) and in academia. Recently, he has earned the award by the National System of Research (Mexico) in recognition of his research labor. At present, he is part of the IMPROWEB project to improve software processes in 32 Spanish enterprises.

Pacheco Carla

She has the PhD studies in Software Engineer and is full-time professor at the Postgraduate Division of Technological University of the Mixtec Region, Mexico. She is author of several international papers related to Software Engineering and, more specifically, Requirements Engineering. She is a reviewer of the International Journal of National Polytechnic Institute of Mexico. Pacheco has participated in several Mexican projects related to software requirements as a part of process to improve the Mexican software industry.

Calvo-Manzano Jose A.

He is PhD in Computer Science. He is assistant professor in the Computer Science School at the Polytechnic University of Madrid. He is teaching in the area of Software Engineering, specifically in the domain of software process management and improvement. He has participated in more than 20 research projects (European and Spanish Public Administration). He is author of more than 50 international papers. He is author of books related to software process improvement and software engineering topics also. He is a member of SOMEPRO research group, where his main research field is process improvement, focusing in the topics of requirements management, project management, process deployment, and solicitation and supplier agreement development.

Cuevas Gonzalo

He received an Engineering degree in Telecommunications in 1965 and a PhD in Telecommunications in 1974. He also received an MS in Computer Science from the Polytechnic University of Madrid in 1972. He has been vice dean of the Computer Science faculty at the Polytechnic University of Madrid, where he is a full professor since 1970. He worked for Iberia Airlines as Software Development Director, supervising over 200 technicians, Data Processing Centre Director, Data Transmission Software Development Director, and the person in charge of strategic planning and software process improvement. He led European projects on software best practices from 1991 to 1995. His main research field is software engineering, including both technology (methods, techniques, and formalisms) and management. His current research interest is process models and methods, and transition packages. He is member of ACM, senior member of IEEE, member of the Telecommunication Engineering Association and member of the Computer Sciences Association.

San Feliu Tomás

He is PhD in Computer Science. He has been working 15 years to Software Engineering field as programmer, associate and assistant professor and consultant. He has been involved in CMM appraisals in Spain and Latin America. As Software Engineering assistant professor at the Polytechnic University of Madrid, his main research field is software process and software process improvement. His current research interest includes project management and risk management. He has published several books on process improvement and he has published technical papers in software engineering and software process improvement.

Mendoza Eloy

He graduated in Computer Engineering in the Technologic Institute of Oaxaca in 2000. He received a MS degree in Software Engineering from Technological University of the Mixtec Region in 2008. He has been professor at two elementary schools and he has developed software for several organizations. His research interests are on software engineering, specifically on software process improvement.