EuroSPI 2008 Proceedings

Proceedings

The papers in this book comprise the industrial proceedings of the EuroSPI 2008 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 978-87-7398-150-4, Publisher Code 7398.

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 15th 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

Conference Chair:	Richard Messnarz (ISCN, Ireland)
Scientific Program Chair:	Rory O'Connor (Dublin City University, Ireland)
Scientific Program Chair:	Nathan Baddoo (University Hertfordshire, UK)
Scientific Program Chair:	Kari Smolander (Lappeenranta University of Technology, Finland)
Industrial Program Chair:	Jorn Johansen (DELTA, Denmark)
Industrial Program Chair:	Mads Christiansen (DELTA, Denmark)
Industrial Program Chair:	Nils Brede Moe (SINTEF, Norway)
Industrial Program Chair:	Risto Nevalainen (STTF, Finland)
Industrial Program Chair:	Stephan Goericke (ISQI, Germany)
Tutorial Chair:	Richard Messnarz (ISCN, Ireland)
Organizing Chair:	Adrienne Clarke (ISCN, Ireland)

Industrial Programme Committee

ALEXANDRE Simon CETIC asbl, BELGIUM

BALSTRUP Bo Center for Software Innovation, DENMARK

BARAFORT Beatrix Centre de Recherche Public Henri Tudor, LUXEMBOURG

CHRISTIANSEN Mads DELTA, DENMARK

GOERICKE Stephan International Software Quality Institute, GERMANY

GRIESSER Martin Continental Engineering Services GmbH, GERMANY

HEHN Uwe method park Software AG, GERMANY

HIND Tim Axa Sun Life, UK

IVANYOS Janos MemoLux Organizing Developing and Enterprising Ltd., HUNGARY

JOHANSEN Jorn DELTA, DENMARK

KISHIDA Kouichi SRA Inc., JAPAN

LICHTENECKER Gerhard MAGNA STEYR Fahrzeugtechnik AG & Co KG, AUSTRIA

MESSNARZ Richard, ISCN, IRELAND

MOE Nils Brede SINTEF ICT, NORWAY

NEVALAINEN Risto STTF Oy, FINLAND

OJALA Pasi O. Nokia, FINLAND

O'LEARY Eugene EQN Ltd., IRELAND

RIEL Andreas VRL KCiP INPG, FRANCE

SEHL Georg iMBUS AG, GERMANY

SMITE Darja RITI, LATVIA

SPORK Gunther MAGNA Powertrain AG & Co KG, AUSTRIA

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

TÜHNCHER Gerhard ZF Friedrichshafen AG, GERMANY

VON BRONK Peter Systemberatung Software-Qualität, GERMANY

WESTERHEIM Hans SINTEF ICT, NORWAY

MIKKELSEN Edda, Det Norske Veritas Software, NORWAY

EuroSPI Board Members

ASQ, http://www.asq.org

ASQF, http://www.asqf.de

DELTA, http://www.delta.dk

ISCN, http://www.iscn.com

SINTEF, http://www.sintef.no

STTF, http://www.sttf.fi

Editors of the Proceedings

Dr Richard Messnarz, ISCN, Ireland

Mads Christiansen, DELTA, Denmark

Damjan Ekert, ISCN, Austria

Welcome Address by the EuroSPI General Chair

Dr Richard Messnarz

EuroSPI is an initiative with 3 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.

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 14th 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 (EQN - EU Leonardo a Vinci network project).

With a general assembly on 15.-16.10.2007 through EuroSPI partners and networks, in collaboration with the European Union (supported by the EU Leonardo da Vinci Programme) a European certification association has been created for the IT and services sector to offer SPI knowledge and certificates to industry, establishing close knowledge transfer links between research and industry.

With the support of the E-Learning EU project PLATO (2005 – 2007) an infrastructure for online learning has been set up for 12 European professions related with SPI.

Since beginning of 2008 a new European initiative EU Certificates Campus (2008 – 2010) has started where key knowledge will be put in online seminars around EuroSPI.

Welcome to Ireland by Kevin Ryan

Prof. Kevin Ryan

Lero

As the Director of Lero - The Irish Software Engineering Research Centre - I am very pleased and proud to welcome the EuroSPI 2008 conference to Ireland. Information and Communications Technology (ICT), particularly software, has played a vital role in Ireland's remarkable economic growth. The Industrial Development Authority (IDA) reports that nine of the world's top 10 software companies have significant operations in Ireland, helping to make us the largest exporter of software in the world. In 2007, Ireland exported €50 billion worth of ICT products and services. Forty percent of the packaged software and 60 percent of the business software sold in Europe originates from Ireland. Ireland now faces the key challenge of sustaining this remarkable development in a changing world.

Recognizing the crucial role that technology would play in Ireland's future as a knowledge society, the Irish government commissioned a national technology foresight exercise in 1998. As part of this effort to ensure Government investments in research, science and technology would, in future, underpin Ireland's economic competitiveness and development as a knowledge society, Science Foundation Ireland (SFI) was established. Lero - The Irish Software Engineering Research Centre - was set up in 2005 with support from SFI's CSET (Centre for Science, Engineering and Technology) programme. Lero is a collaborative organisation, initially involving researchers from four universities: the University of Limerick, Dublin City University, Trinity College Dublin, and University College Dublin. Lero researchers work in five interrelated areas: global software development, autonomic systems, service and aspect-based architectures, mathematics for software engineering and software product lines. Lero's has a particular focus on automotive systems, medical devices, and financial/enterprise systems - domains that have great potential for Ireland.

As software engineering is, by definition, an applied discipline, Lero's research agenda is informed by the requirements of its chosen industrial domains. The researchers concentrate on problem areas that have potential real-world application and much of the research is carried out in collaboration with industry partners. Cooperation between universities, industry, and academia are fundamental and fruitful in helping to maximize the impact of necessarily limited resources. The Irish software engineering research community welcomes EuroSPI to Ireland and we look forward to working with EuroSPI conference delegates now and in the future.

Welcome to Dublin City University by Rory O'Connor

Dr. Rory O'Connor

Local Chair

As the local chair of the EuroSPI 2008 conference I welcome you to my home city of Dublin and to Dublin City University in particular. Dublin, or *Baile Átha Cliath* in the Irish language, has been at the centre of Ireland's exceptional economic growth over the last 10-15 years, a period referred to as the *Celtic Tiger* years. Central to this competitiveness and development of Ireland has been the vision of Ireland as a knowledge society, driven by education, research and development. Dublin is the primary centre of education in Ireland, with three universities and a total of 20 higher education institutions in the city, with the oldest dating back to the 16th century.

Dublin City University (DCU) was initially created as the National Institute for Higher Education in 1975 and it enrolled its first students in 1980, before being elevated to university status in 1989. Like other universities, DCU is a place of learning, but in a rapidly changing world not all universities are the same and there is an increasing diversity of mission. DCU's focus is unique in Ireland: it sets out to develop high quality, high value learning within the wider setting of Ireland's economic and social needs. It ensures that students gain direct experience of industry and other workplaces, and it offers the very latest technology and facilities to ensure that all are equipped to become leaders in their chosen fields. This university is also committed to the further development of its commercial activities, with a growing emphasis on technology transfer and the establishment of campus companies, which exploit commercially the discoveries and skills of the campus community.

On behalf of DCU I welcome EuroSPI 2008 conference delegates to both Dublin and DCU, and wish you a successful and enjoyable conference. If you can find the time in your schedule I highly recommend a visit to Dublin city centre. Dublin offers its visitors a wealth of attractions to explore, from Churches, Historic Buildings, Libraries, Museums and Galleries, and Parks and Gardens. Of course there is also Dublin's famous vibrant nightlife!

Contents

Experience with Innovation Checks: A Case Study with 46 Companies in Denmark

Jørn Johansen and Mads Christiansen
DELTA Axiom
Venlighedsvej 4, DK-2970 Hørsholm, Denmark
joj@delta.dk and mc@delta.dk,
www.deltaaxiom.com

Abstract

During a two year period DELTA has performed 46 Innovation Checks in Small and Medium-sized Enterprises (SME) with great success. In addition to reported benefits experienced by the companies, the Innovation Checks have given us extensive knowledge of the companies' difficulties in having success with innovation in relation to the following 5 topics: Product, Processes, Production, eBusiness (use of IT for business proposes), and Marketing.

This paper presents the method behind the Innovation Checks and conclusions reached from this important project carried out in cooperation with DI (the Confederation of Danish Industry).

The main conclusions are:

- Companies don't have the necessary time to invest in innovation
- Companies lack a business strategy that includes innovation
- Without the time and focus, innovation will be ad-hoc rather than a mastered discipline
- An Innovation Check does help companies with innovation here and now, but without a mastered innovation process it is likely to become a one-off event.

This paper presents our experience from data and observations collected during the Innovation Checks. As the population is fairly small (46 companies) the results are to be taken as observations and guidelines rather than scientific facts.

Keywords

Maturity, product innovation, process innovation, production innovation, marketing innovation, eBusiness innovation, SME.

1 Introduction

The Danish Employers' Association (Provinsindustriens Arbejdsgiverforening (PA)) with more than 800 company members expressed a wish to support their members to become more innovative and competitive. Thus, they decided to invite their members to apply for an Innovation Check, 50 checks in total sponsored by PA. ITEK (the Danish ICT and electronics federation for it, telecommunications, electronics and communication enterprises under DI) and DELTA were assigned to develop and implement the Innovation Check concept. The project name is Intelligent Steel, which signalizes the introduction of innovative, intelligent products within the steel and metal industry.

The overall goal is to encourage the member companies to focus more on innovation and become more competitive through innovation, and the success criteria are:

- All companies will report new ideas for innovation activities
- More than half of the companies will initiate one ore more new innovation activities based on the Innovation Check.

The project was launched in February 2005 until November 2007. During that period, 52 companies applied for an Innovation Check and actually 46 Innovation Checks have been completed. This paper is based on data input from the 46 Innovation Checks.

The participating 46 companies represent a very heterogeneous group with a number of employees ranging from 13 to 1000. Their line of business varies significantly: road signs and control panels, beverage dispensers (e.g. for draught beer), greenhouses, steel and metal work, diesel engines for boats, pumps, transport and logistics, cranes etc.

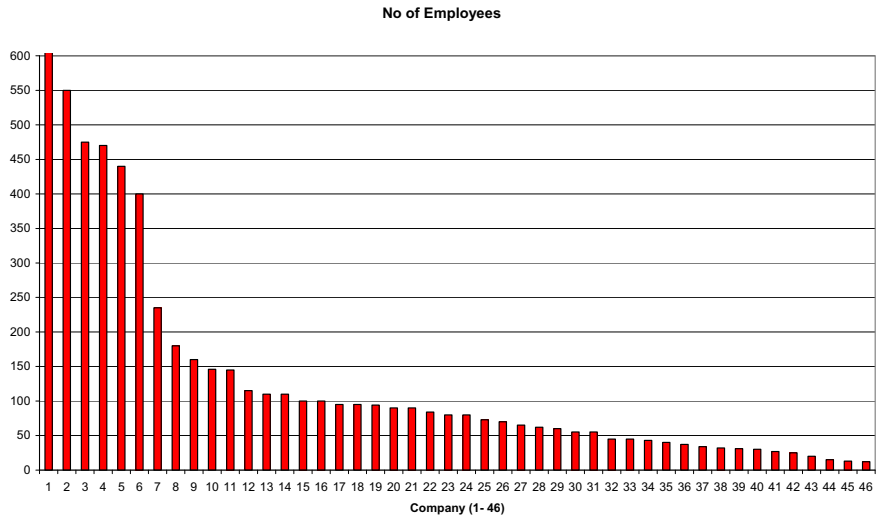


Fig. 1 - The distribution of employees in the 46 companies

2 The Innovation Check concept

When designing the Innovation Check concept it was important to:

- Provide the participants with a positive and beneficial experience
- Involve senior management at the highest organizational level in the company
- Involve the right competences, external as well as internal.

The data collection during the Innovation Check was a great opportunity to conclude on the effect and to improve the concept in becoming a standard SME service for the future.

We have developed a 3 phase model which will be presented below.

2.1 Phase 1 – Identification of ideas for innovation

The Innovation Check starts with a kick-off meeting in order to inspire the company to focus on potential and relevant areas for innovation, whether product, process, production, eBusiness or marketing innovation. The first meeting that takes between 5 and 8 hours and involves the top management and selected key employees of the company, is a mix between interview and discussion of the company's actual conditions.

All five innovation topics are covered by external senior experts, who also attend the meeting:

- A product and production innovation expert from DELTA (actually the CEO)
- A process and process innovation expert from DELTA Axiom
- A business and marketing innovation expert from DI

An additional benefit from these meetings was the presence of DELTA's CEO. The implementation of any changes usually needs the support and sponsorship of the CEO – therefore the CEO has to be present. However, CEO's are often very busy and reluctant to spend an entire day on a meeting, especially if they consider it a nice offer rather than a necessity. If DELTA's CEO could invest a full day it was easier to convince the companies to turn up with their strongest team as well.

The meeting will be followed by a written 20 page report including approx. 30 new ideas for innovation. The final topic on the agenda is to select two ideas for further processing during Phase 2.

2.2 Phase 2 – One or two days expert assistance

The processing of the selected ideas usually requires expert assistance to provide sufficient knowledge of e.g. new technology, project competence, maturity, user driven innovation, eBusiness solutions, before any decisions can be made. During Phase 2, one or more experts arrange a meeting with the company and setup a proper agenda and timeframe. The idea is not to implement a solution (which can't be done in such a short timeframe anyway), but to provide more information about the selected issues in order to be able to make better decisions. The experts were typically experts in application, process, business or development from DI, DELTA and their network. It was not that important to promote DI and DELTA as it was to promote our knowledge of how to find any expert within our network. Normally, the Development Manager and key employees participated in these meetings, the top management as well when relevant.

Depending on the selected ideas, a one or two day programme was designed with various activities e.g. seminars or workshops with topics such as innovation methods and techniques, project management, sensor technology, selecting the right microchip design solution, wireless technology, eProducts (i.e. putting products on the Internet), marketing, maturity and process improvement, light maturity assessment, intranet demonstration, *ImprovAbility*™ [1], and LEGO Serious Play [2].

During Phase 3 the existing report was added with another chapter concerning achieved results, knowledge and experience.

2.3 Phase 3 – follow up and closing the Innovation Check

The idea of this phase is to support the companies with an "external pressure" urging them to continue their innovative initiatives, to present the report and to close the innovation check with a follow-up on the entire sequence of activities.

This meeting lasts for approx. two hours with the CEO and employees involved as well as one representative from DELTA.

A questionnaire with both quantitative and qualitative data is completed in an interview form. These data form the basis for this paper.

3 Example Cases

Company A is a manufacturer of sewer cleaning equipment e.g. slurry pump vehicles. The company is family owned and established in 1915. The number of employees is 115 and they export to 28 countries with an export rate of 77%. They consider themselves a market leader.

The Innovation Check came up with the idea of adding a number of sensors, using GPS technology and putting the slurry pump vehicle on the Internet. The sensors are used to measure e.g. toxic content and send data to a database with information on time and place. Thus, the company has been transformed from a sewer cleaning equipment manufacturer into a complete environmental service provider. The GPS provided a tool for better route planning allowing the customer to measure the driven route.

Adding new measurements to the existing PLC provided the possibility to see number of revolutions and running hours for the high pressure pump and the vacuum pump in order to benchmark vehicles and optimize usage. Finally the system introduced the possibility of transmitting time sheets to the ERP system at the office.

Company B develops and produces robot based systems for e.g. welding machines to garages, light industry and all-round welding machines. The company is established in 1970, has 40 employees and an export rate of 65%.

The innovation check came up with the idea of using wireless communication technology to monitor the robot and welding machines. Data from the automated welding process are transmitted to a handheld PDA via the wireless internet. If a process measurement exceeds preset limits, or the process is running out of material etc., the operator automatically receives an alarm on his PDA.

4 Results

Phase 3 includes interviews with the management about general business information relevant for

innovation; e.g. turnover, number of employees and percentage of export and process focus. Based on the data material we will present some interesting findings below:

Table 1: Process focus versus innovation and export.

		Existence of process focus			
Number of employees	**Data**	**Low**	**Medium**	**High**	**Grand Total**
Medium (> 50)	Importance of innovative products	2,00	2,20	2,64	2,32
	Development cost in % of turnover	1,7%	3,1%	5,0%	3,6%
	Number of new products with innovative elements	1,60	1,93	2,40	2,03
	Average of Export in %	49%	76%	68%	69%
Small (< 50)	Importance of innovative products	2,00	2,11	2,67	2,20
	Development cost in % of turnover	5,2%	2,6%	7,0%	4,1%
	Number of new products with innovative elements	1,67	2,00	2,67	2,07
	Average of Export in %	27%	41%	66%	43%

Importance of innovative products: Unimportant: 0, Less Important: 1, Important: 2 and crucial: 3
Number of new products with innovative elements: <10%: 1; 10%-25%: 2; 25%-50%: 3; >50%: 4

Table 1 illustrates the correlation of innovation activities and export to process focus in the organization. Some comments on the data:

1. Regardless of company size, the existence of process focus increases with the importance of making innovative products. In other words, there seems to be a good correlation between focus on innovation and process – innovative companies also have focus on processes. The number of new products with innovative elements also increases with the existence of process focus.
2. Small companies spend a higher percentage of their turnover on development than larger companies. This seems fair, since development costs represent an invariable element and turnover is increasing with company size.
3. Small companies with little focus on processes are likely to be local actors with only limited export.

Table 2: Innovation importance versus process focus, innovation and export
(Note: small limit 50 employees)

		Importance of being innovative			
Number of Employees		**Less important**	**Important**	**Crucial**	**Grand Total**
Medium (> 50)	Process focus	1,50	2,23	2,36	2,19
	Number of new products with innovative elements	1,25	1,82	2,43	2,03
	Average of Export in %	74%	75%	61%	69%
Small (< 50)	Process focus	1,50	2,00	2,20	2,00
	Number of new products with innovative elements	1,00	2,13	2,40	2,07
	Average of Export in %	2%	48%	53%	43%

Process focus: Low 1; Middle 2; High 3
Number of new products with innovative elements: <10%: 1; 10%-25%: 2; 25%-50%: 3; >50%: 4

Table 3: Innovation importance versus process focus, innovation and export
(Note: small limit 100 employees)

		Importance of being innovative			
Number of Employees		**Less important**	**Important**	**Crucial**	**Grand Total**
Medium (> 100)	Process focus	2,00	2,50	2,71	2,57
	Number of new products with innovative elements	1,00	2,00	3,00	2,46
	Average of Export in %	75%	84%	65%	74%
Small (< 100)	Process focus	1,40	2,00	2,08	1,94
	Number of new products with innovative elements	1,20	1,93	2,08	1,87
	Average of Export in %	45%	57%	56%	55%

Process focus: Low 1; Middle 2; High 3
Number of new products with innovative elements: <10%: 1; 10%-25%: 2; 25%-50%: 3; >50%: 4

It appears from Table 2 and 3, which are basically the same apart from different limits in company size, that:

1. Small companies (< 100 employees) have less process focus in average (1.94) compared to larger companies (2.57). If small companies have less than 50 employees, process focus is less evident. This leads to the conclusion that process focus becomes interesting when companies reach approx. 100 employees. The exact number is not derived from our material due to the fact that the number of companies is quite small and the group is very heterogeneous.
2. Process focus increases with the importance of being innovative regardless of company size. The question is, however, what comes first: Are process focused companies more innovative or do innovative companies have to focus on process to be innovative. Or is it a combination?
3. Small companies have a lower export rate than larger companies. Being on the export market is very demanding, but there is a tendency towards small and more innovative companies having more success with export. Looking at the two tables, it seems as if the very small companies with very little focus on innovation will not succeed with export (only 2%). These data are only based on 2 companies, which is why conclusions cannot be drawn without more data.
4. Fortunately, there seems to be a good relation between companies saying that innovation is important and the number of products with innovative elements they develop.

We have not been able to see any correlation between innovation and turnover, profit and profit margin nor have we seen any correlation between process focus and turnover, profit and profit margin.

Some answers to the qualitative part of the phase 3 interview are found below.

Question 1: What is most important to focus on when innovation is the goal?	
Top 10	**Percentage**
Employees (competences and involvement in innovation)	57%
Product development (increase investment and improve)	50%
Production technology (increase investment and improve)	46%
Customers (involvement in innovation)	26%
Partners and suppliers (involvement in innovation)	22%
Management (involvement in innovation)	22%
Finance (internal financing and new external funding possibilities)	13%
Market and marketing (possibilities)	9%
Product technology (investment in new technology for new products/solutions)	4%
Logistic (increase and improve)	4%

Table 4: Percentage of companies (management) addressing the topic as an important focus for innovation (text in brackets represents the authors' interpretation of the addressed topic).

Observation 1: Employees, development and production are essential

Management ranges employee's competences and involvement, product development and production technology as the most important focus for innovation. It seems natural because these are the basics for running a company – effective development and production of products executed by competent employees. **But** it is a bit of a puzzle why topics as business development and strategy are not mentioned at all, perhaps because the companies are not used to work with these?

Observation 2: Involvement of external and internal relations is important

It is a fact that involvement of people and competences both internal (employees and management) and external (customers, partners and suppliers) is important. External experts e.g. designers, business consultants and other consultants, however, are not considered necessary for innovation. Companies seem to believe that they have all competences they need internally or in their closest network.

Observation 3: Market and financing are not important issues

Surprisingly it seems as if the companies don't consider knowledge of the market as an important topic for innovation. Some markets are conservative and develop slowly, which makes the companies consider their knowledge of the market to be sufficient. It is worrying that knowledge of a market is not considered important for innovation. Companies do find their customers important for innovation, i.e. they can make innovative solutions for the specific customer. It seems more complicated, however, to innovate for the broader market. Finance does not seem to be a big issue.

Observation 4: New technology is not an innovation driver at all

New technology has never been as multifarious as it is today. Every day brings several new technological solutions, however management does not see this as a focus area for their innovation and in use for new innovative products. In this study, new technology is really the last topic companies think of.

Question 2: What is important for establishing an innovative environment?	
Top 10	**Percentage**
Management commitment (use time and resources on innovation)	57%
Creative employees (able to find solutions)	41%
Interdisciplinary collaboration (across professions and departments)	33%
Open about problems (we can learn and innovate from problems in general)	28%
To have time and resources (which is a problem in day-to-day work)	20%
Collaboration with customers (solutions to their problems = innovation)	17%
Knowledge and knowledge sharing (Interdisciplinary)	15%
Other (such as culture, readiness for change, economy and facilities)	13%
A reward (new solutions or ideas will release a reward)	7%
Confidence (in general)	4%

Table 5: Percentage of companies' (management's) addressing the topic as important for the establishment of an innovative environment (text in brackets represents the authors' interpretation of the addressed topic).

Observation 5: Management commitment is essential

It is important and positive to notice that management acknowledges the importance of their commitment to fund and support an innovative environment and culture in the company. They know they are responsible and without their support, attention and involvement innovation will not have a chance.

Observation 6: Collaboration, creativity, knowledge sharing, time and resources are important

Interdisciplinary collaboration between creative employees and knowledge sharing are identified as important. This sounds logical, but it does not fit well with the lack of time and resources having the most restrictive influence on innovation (from question 4).

Observation 7: Rewards are not considered important

Management does not consider it important to reward employees when they provide new and innovative ideas. We cannot see if the employees agree or would consider a reward as an argument to put forward more innovative ideas, since only the management group has answered this question.

Question 3: Where do new innovative products, solutions and ideas come from?	
Top 5	**Percentage**
Customers (initiated by collaboration or solution to their problems)	91%
Employees (initiated by solution to problems – products or customer problems)	74%
Partners and suppliers (initiated by collaboration or solution to problems)	28%
Competitors (ideas from their products)	24%
Other (e.g. from management, market, authorities or salespersons)	22%

Table 6: Percentage of companies' (management's) answers to where the innovative products, solutions and ideas come form (text in brackets represents the authors' interpretation of the addressed topic).

Observation 8: Costumers and employees are the source to new innovative products

Customers and employees are acknowledged as the most important sources for new innovative products, solutions and ideas. Ideas from partners, suppliers, competitors and management are considered equally but less important. The reason why might be that innovation is seen more as a solution to problems than a strategic basis for the future of the company.

Question 4: What are the most restrictive influences on innovation?	
Top 9	**Percentage**
Lack of time and resources (daily work and daily problems prevent innovation)	39%
Lack of qualified employees (no one to hire)	28%
Lack of knowledge of market and technology (weak basis for decisions)	22%
High costs and lack of capital (lack of capital for investment)	17%
Market resistance (market does need the product)	17%
Uncertainty about investment (risk in relation to investment in new products)	15%
Other (such as wrong or missing strategy, product complexity, politics and taxes)	11%
Internal resistance (new requirements, new tasks and maybe change of job)	9%
Risk (in general)	7%

Table 7: Percentage of companies' (management's) addressing the topic as the most restrictive on innovation (text in brackets represent the authors' interpretation of the addressed topic).

Observation 9: Lack of time and resources block the way for innovation

Lack of time and resources are the most frequently mentioned reasons for lack of innovation effort in the companies. A well-known scenario is that the daily work (e.g. upcoming problems) takes all spare time allocated for innovation because fire extinguishing is considered most important – the dilemma being what is urgent and what is important. "If only we had enough time and resources (creative and fiery souls), then we would have the time to be really innovative".

Observation 10: It is difficult to pinpoint the most restrictive influences on innovation

These questions are answered less strikingly than the other ones. The answers are scattered and it is difficult to generalize, perhaps because innovation is not in focus and management therefore has not reflected sufficiently on the reason why innovation is limited.

Observation 11: Risk is secondary – lack of basis for decision making is primary

Risk both in relation to investment and to other type of risks (success with development, use of technology, market resistance) is well acknowledged. But is this the main reason for mentioning risk as a restrictive influence on innovation? The risks could be derived from a more considerable risk – lack of basis for decision making in relation to innovation investment.

Observation 12: Outsourcing / Insourcing is missing

Many companies talk about outsourcing (and some also about insourcing) during the Innovation Check. The companies did have some good, but mostly bad experiences with outsourcing. However, data show that no company sees outsourcing as part of innovation. Strangely enough, outsourcing is often regarded as an innovative way of improving business (production, subsuppliers and even development), but not in this context.

The sourcing problems have led to a new 3 year project "SourceIT", where Roskilde University (RUC) and DELTA together with three companies will carry out research into companies' ability to innovate in a context with optimal outsourcing.

5 Conclusion and Discussion

It seems obvious that management typically is very open and favourable to innovation, especially if innovation is related to solving problems and improving existing products. When it comes to new products, is seems easier to fit them into the existing product range rather than to try to expand business with new and innovative products.

Many companies seem to lack a business strategy that includes innovation. The excuse for lack of innovation is lack of time and resources with the right competences, lack of knowledge of market and technology as well as high costs. The real problem, however, seems to be the lack of a business strategy that includes innovation.

Being present on conservative and/or slowly developing markets may lead the companies to rely on employees getting a good idea or on customers requesting solutions that require innovative thinking as the primary sources of innovation. Developing innovative new products based on market knowledge, market needs and new technology, which can introduce new business areas such as services, seems more difficult. In most companies the presence of innovation as a systematic discipline including market needs and technology opportunities for developing new businesses is non-existing.

A questionnaire was sent to all participating companies upon completion of the Innovation Check. The result of this survey shows that

- 82% of the companies have benefited form the Innovation Check
- 86% have got new ideas
- 73% have initiated one or more of the recommended innovative initiatives.

Comparing this to the project goals makes the Innovation Check a very successful project. However, it is unlikely that the companies will participate, if they have to pay for the Innovation Check themselves. The Innovation Check has helped, but only this time. Without a business strategy that includes innovation and a mastered innovation process, we doubt that innovation in these kinds of businesses will ever be more than incremental and occasional.

6 References

[1] Pries-Heje, J. and Johansen, J., "Improve IT, A book for improving software projects", DELTA, 2007.

[2] http://www.seriousplay.com/

Author CVs

Jørn Johansen, DELTA

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 13 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

Mads Christiansen has an M.Sc.E.E. from DTU (Danish Technical University) and more than 29 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 10 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. Mr. Christiansen is also the ´coordinator of a Danish knowledge exchange group: User Centered Design - Developing user friendly products.

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

Changing Attitudes – Improving efficiency

Dominic Michell, Serco
and
Jill Pritchet, Software Measurement Services Ltd

Abstract

This presentation is based on the real life experience within Serco of a major Process Improvement Programme, which began in February 2006 and is continuing today. The focus was not only on retention of the existing ISO 9001:2000[1]/TickIT certification, but also to provide a framework which would be used to gain ISO 20000:2005[2] Certification at a later date.

The presentation provides first-hand experience from those directly involved. By sharing the experience with others we hope that we can give the audience things they may wish to consider if they are involved in a similar undertaking.

The presentation will be looking at the more human side of change and how the participants were both affected by, and reacted to, the problems and challenges they were faced with.

Those attending the presentation will benefit by gaining an insight into the benefits and pitfalls of creating a staff driven improvement programme and the achievements that can be realised.

Keywords

Change Management, Continuous Improvement, Culture, ISO 20000:2005, ISO 9001:2000, Management Commitment, Metrics, Motivation, Process Improvement, Service Management, Software Development Life Cycle and TickIT.

1 Background

Serco is a medium sized software house providing services to the UK Government. Their work is performed within customer contracts and provides systems for business intelligence and workflow. Serco Technology is part of the larger Serco Group.

Software Development is Object Oriented using:

- Java 5 with Oracle back end database.
- C and C++
- Rational Rose
- Eclipse
- MKS configuration management tool

Serco runs a Windows platform operating on data centres with Blade technologies, LANS and WANS. The projects undertaken by Software Development use teams of between 5 and 24 Analysts and Developers and in value terms range between £000's and up to £1.8M.

1 I.S. EN ISO 9001:2000 Quality Management Systems – Requirements

2 ISO/IEC 20000-1:& 2 2005 Information Technology Service Management – Part 1 Specification & Part 2 Code of Practice

2 Introduction

Serco was facing a number of challenges in 2006; not only maintaining the good relationships with their existing client base but also significant expansion of the business to take on new high profile clients. They needed to have absolute confidence that their existing Software Development and Delivery could cope with the major changes that were planned.

In February 2006 Software Measurement Services (SMS) were commissioned to perform an assessment of the software development activities at Serco, from bid stage to post development support. The objective was to compare Serco to industry best practice and provide a benchmark for starting a continuous improvement programme.

The SMS report identified a number of areas that were considered to be at risk and could even jeopardise Serco's chances of passing their ISO 9001:2000[3] re-certification audit. The areas highlighted were:

A programme of projects to improve the software development capability was initiated in March 2006 and the staff called it PRISM.

Figure 1: PRISM **Process, Review, Implement, Standardise, Maintain**

3 The Consultant's view

3.1 Methodology Followed

A specific methodology was not being followed; in reality a common sense approach was adopted. However, looking back it is possible to see that the project did closely follow the Ideal Wheel[4] methodology even though this was unintentional. The driver was to get to industry best practice rather than a specific methodology. We also adopted the Plan, Do, Check, Act methodology[5] throughout the Prism project.

[3] I.S. EN ISO 9001:2000 Quality Management Systems – Requirements

[4] Bob McFeeley, Ideal[SM]: A Users guide for Software Process Improvement, Feb 1996

[5] W, Edwards Deming, The Deming Management Method, 1986

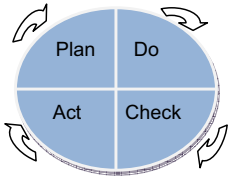


Figure 2: PDCA Cycle

It was always felt that the approach had to be one that would work in the Serco environment and therefore it was adapted at each stage to suit the client's individual requirements.

One of the main areas where we differed from the steps of the Ideal Wheel was in terminology e.g. what they call 'Process Action teams' we called 'Streams'.

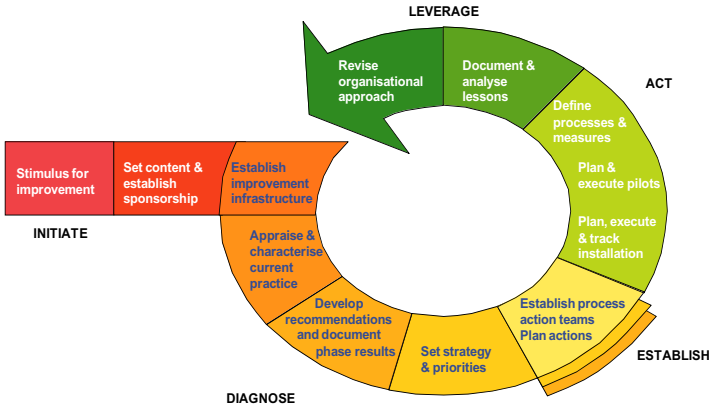


Figure 3: IDEAL WHEEL

3.2 Change Programme Approach

From the very start the decision was taken that the work would be led by the Development Staff themselves. A series of work streams were set up and each stream used the SMS reports findings to review current practices and suggest improvements.

The term 'stream' was chosen to differentiate between their normal day to day activities and work specifically done within the PRISM project to make improvements to their working practices in line with the requirements of the ISO 9001:2000[6] standard.

The Work Streams were made up of staff volunteers representing a cross section from every Software Development team. The initial streams, based on the priorities within the SMS report were:

[6] I.S. EN ISO 9001:2000 Quality Management Systems – Requirements

These streams were selected because:

- They were where the most weaknesses had been found.
- They aligned with the organisation's goals.
- They would help towards the goal of ISO 9001:2000 retention.
- They were aligned to the people's capabilities at the time.

The communications stream covered all streams and co-ordinated the communications with the Steering Group, the Software Development Department and external departments (who were not directly involved in the project).

Work associated with improvements in the Testing area were outside of scope for the PRISM project and were therefore handled in respect of the interfaces to and from the external department to better control handovers etc.,

Individual Stream Leaders were put in charge of each Work Stream and those taking this role were deliberately not the senior people within Development. They co-ordinated the work and ensured that views were sought from a wide selection of people, at all levels within the Organisation before a proposed solution was put forward. The streams were continually supported in their activities by both Serco Management and SMS.

In addition, a Steering Group was set up chaired by the most Senior Manager at Serco. This group met monthly and the stream leaders attended and presented their achievements. This often resulted in further iteration of the proposed solutions based on feedback given at the meeting.

The proposed solutions were formally documented using a pre-defined standard process model approach and piloted before roll out to the wider organisation. The Steering Group had to give formal sign off before roll out could commence.

Training in the form of both awareness events and training courses for those required to use the new processes were arranged.

3.3 Why this approach was chosen

The Work Streams promoted an inclusive approach which helped to gain staff commitment and encouraged staff ownership to the change programme.

Clear goals and targets were set for each Work Stream to ensure everyone was clear on their overall responsibilities, priorities and objectives. This created an early Shared Vision with everyone pulling in the same direction.

3.4 What worked well

Creating a separate Communication work stream to co-ordinate the information from all the work streams was very effective. Regular Newsletters, Intranet notices and emails meant that the work was visible within Software Development as well as to other external teams and clients.

The streams were asked to provide visual displays of their results, as they progressed the work e.g. Process Model diagrams and associated template documents. This acted as a catalyst for discussion and led to further improvements across the operation.

The development of the Software Development Lifecycle (SDLC) proved to be the most important element. The mapping that was initially done, individually per work stream, was gradually brought together and the impact this had when people could see how their part fitted with the others was quite remarkable.

Some of the individuals who took on the role of Stream Leaders changed significantly during the nine

months of the programme. Some started out with feelings of uncertainty about the task and how realistic it was going to be to complete it alongside their day to day activities. Over time they realised that it was possible to manage them both.

In addition, the programme has helped the Software Development Manager to identify which staff members may be ready to take on more senior roles e.g. to move into Project Management.

3.5 What didn't work so well

The Metrics work stream was badly affected by the availability of the Stream Leader who kept getting taken off to do other tasks and also the lack of experience with metrics. Unfortunately momentum was lost and there was an inability to identify suitable alternative resource to take this stream further. This has had a direct impact on the ability to provide quantifiable results for the project as the planned data to show the before and after position was not produced.

Work on Document Control was also impacted because the Stream Leader left the company part way through the project. Someone else took over the work but was unable to devote the time necessary to achieve the expected results. Imminent implementation of a tool Q-Pulse to help with document control also affected the progress.

Work on Project Management and Programme Office was placed lower down the priority list as there was limited in-house expertise to manage the necessary changes and to run the existing projects at the same time.

3.6 Adapting the approach to ensure success

During the change programme it was important that the team remained flexible and responsive. Not all the proposed changes worked first time and we continually monitored progress and reviewed our approach. Where necessary, changes to the approach were made, including in some instances changing a stream leader who had recognised they did not have the skills needed to convince their colleagues that change of this sort was worthwhile. There is no doubt that some people are very good at this and others find it much harder.

One of the benefits was an appreciation within the teams that 'managing' is not always as easy as it might seem. They now have a very different perspective in tackling their current role and to the process of change.

It was recognised that without visible management commitment at the most senior level the programme was likely to fail. Once the Senior Manager agreed to chair the Steering Group, and he attended every meeting, we knew we had to make his valuable time commitment worthwhile. This was achieved by adapting the format of each Steering Group meeting so that there was always something fresh and interactive. The Stream Leaders were presenting what they had done face to face with the Senior Manager and getting his direct feedback. It almost became a competition between the streams to have made the most progress. The results were always impressive.

3.7 Software Process Improvement – how improvements happened

The table below illustrates the effect these improvements had on the efficiency within Serco at the time Prism was implemented; the effects are continuing to evolve today.

Before Prism	After Prism	How this improved efficiency
Ad hoc processes which were not fully documented.	Complete set of agreed processes in use.	Consistency in how people work with built in controls. Allowed better relationships with clients and a clearer understanding of each other's expectations and dependencies. Fewer arguements about what process should be followed.
Document templates were not standardised.	Majority of core templates standardised with ongoing. development of additional templates	Consistency in document production. Client see's the same style for documentation.
Reviews of documents and project deliverables ad hoc.	Core documents subject to peer review and formal inspections[7/8].	Defects found and corrected much earlier and less effort on re-work. Actual number of defects now known, prior to Prism this was not the case.
Informal estimation.	Defined approach with COSMIC based spreadsheet tool to calculate estimates.	Process and tool which enables more accurate estimating to be carried out. The tool is gradually being refined as more projects use it.
No Function Point counting used.	Training in function point counting for selected staff and work progressing on use of counts on selected projects. CAST software assessment tool implemented which give actuals vs. Estimates.	Selected staff were trained in Function Point counting so that they could perform counts at an early stage in the projects. Recent implementation of CAST will provide more data on project results going forward.
Silo's very much in evidence with no real 'team' working together to a single goal.	Regular communication between all parties and much more willingness to look at things from another's perspective.	Easier attainment of project goals.
Prince 2[9] used by some Project Managers but not mandated.	Prince 2 aligned with the Software Development Life Cycle and now mandated for all projects.	Improved governance of projects.
Immaturity of those working in software development which impacted how they approached their tasks.	More experienced and confident staff able to work more efficiently with a greater awareness of 'the bigger picture'.	Much more effective communication with the customer and internal departments.

7 M. E. Fagan, 'Design and code inspections to reduce errors in program development', *IBM System Journal*, (1976).

8 M. E, Fagan, 'Advances in software inspections', *IEEE Trans. Software Engineering*, (1986).

9 Office of Government Commerce (OGC) Publisher: Stationery Office 'Managing Successful Projects with PRINCE2 ™ Manual 2005' - 5th impression 2007

4 The client's view of the Programme

4.1 Managing the change programme

One of the key concerns staff had about implementing a change programme was the lack of time available for the additional tasks and a fear that the day to day work would suffer. Management demonstrated its commitment to the programme by agreeing that development staff could have up to 25% of their time allocated to the streams of work. Resource allocated to the change programme was monitored using cost codes and although some individuals did use 25% of their time, the overall average was much lower than expected at only 3.8%.

In the beginning there was a tendency for the streams to get side tracked and spend too much time discussing interfaces between the streams. The Steering Group had to step in and re-focus them on their individual activities. Later on when each stream had developed their own process models and the whole thing was put together on the Board Room wall there was a sharp intake of breath when they realised that many of the connections were already there.

4.2 What skills the staff obtained from the experience

There has been a significant impact on the skill levels of those involved. Not only do they now understand how difficult it can be to change within an organisation but they also understand their colleague's perspectives more deeply. A solution that might work well in one area can in fact have a detrimental effect elsewhere. Previously, they might have just gone ahead anyway and not have been concerned about the impact to others; now they are far more likely to put relevant people together to discuss and jointly agree the approach that best suits them all.

In the past defects were allowed to proceed further into the development activities with an expectation that testing would 'pick things up'. Now people are adapting to regular reviews as a mechanism for finding errors earlier and fixing them. Initially this was seen as 'big brother' but quite quickly people realised that they were involved in less re-work and recognised the benefits of 'get it right first time, every time'. People are now happier to embrace change within the organisation and look to the positives, rather than perceive change as a threat to the status quo.

4.3 The effects on customer relationships

The use of visual displays of work has had a significant impact on the levels of understanding across the Organisation. Having a defined Life Cycle which can be shown to customers has made communication easier. The client's now have a greater understanding of the complexity of managing what they see as a 'simple change request' and a more 'lean approach' is evolving both internally and externally.

Management have more confidence that Software Development activities are properly controlled and this has also had an impact on the Bid cycle. Those seconded to the Bid Teams are now properly equipped to handle the task by having a defined process to follow.

4.4 Unexpected results and benefits

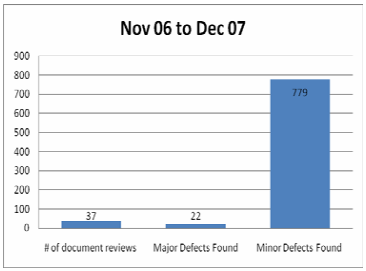
There has been a visible change in the attitudes of both Staff and Management. They both have a better appreciation of their respective roles and responsibilities within the Organisation.

It was not always the people we expected who rose to the challenges presented by the programme.

5 The Results

In a relatively short time significant success has been achieved, the most tangible results being:

- Retention of ISO 9001:2000[10] certification.
- Significant cost reduction for Software Delivery; the first 6 figure project to use the full end-to-end life cycle came in on time and under budget by 29%.
- Defect fix rates have vastly improved and previously unreported defects are now being identified with 95% of bug fixes now happening 1st time i.e. no iteration needed. The bug fix cycle is much more controlled.
- Reviews pick up documentation errors much earlier in the lifecycle, thus reducing development costs.



In addition, the following has been achieved:

- Greatly improved governance
- Reduction in “Silo” working
- The software development lifecycle has been reviewed and documented. Gaps and duplication have been identified and work is underway to resolve these
- People now understand exactly where they fit in the lifecycle, and more importantly, conflicts have been identified and processes updated
- Review checkpoints positioned throughout the SDLC
- Prince 2[11] guidelines positioned through the SDLC with more clarity
- Basic metrics have been established to monitor where defects occur throughout the lifecycle
- The IMS (integrated Management System on the intranet) is being improved and simplified to contain all of the new processes
- Process mapping has been expanded outside Software Development to include a number of other departments, i.e. Service desk
- Induction training now uses the defined processes to bring new starters up to speed quicker

6 Recommendations

The reason for submitting a paper of this sort was to give others the benefit of our experience. Having gone through such a significant change project at Serco there are lessons we have learned along the way which we would like to share.

[10] I.S. EN ISO 9001:2000 Quality Management Systems – Requirements
[11] Office of Government Commerce (OGC) Publisher: Stationery Office ‘Managing Successful Projects with PRINCE2 ™ Manual 2005’ - 5th impression 2007

- Don't be disheartened if things don't come together first time, this is normal and should be seen as an opportunity to achieve a better, longer lasting result. You learn as much, if not more, from the things that go wrong.
- Consider an assessment against best practice rather than a specific model as a starting point.
- Remember, many of the fundamental building blocks will be the same regardless of model.
- Focus on 'Quick Wins' as this encourages people to continue when they see positive results.
- Get the staff that will be using the processes to develop them.
- Make sure you have visible management commitment; our Sponsor was very much part of the process and staff were more committed as a result.
- Setting up a metrics framework should be an early objective so that it is possible to show the results of the change project in a quantifiable way.

7 Conclusions

Whatever approach you select as the basic framework for your own software process improvement initiative there are a number of key factors that will help you to succeed:

- Selection of a 'Champion(s)' who believe in what you are doing and will fight to see it through.
- Regular meetings to applaud success and resolve issues.
- Realistic timeframe for the overall change project.
- Identification of 'Quick Wins' so that there is early success to build from.
- Visible Management commitment.
- Visibility of progress towards your goals is essential. Some of the work will take time and there can be a tendency for staff to become disillusioned if there seems to be no visible progress.

Change programmes involve people and therefore you must ensure you focus on Cultural Change as part of the initiative. There will always be sceptics, the skill is in finding ways to bring them on board, and often led by those who find adapting to change easier.

It doesn't stop here though; you still need to continually adapt and change, the difference is that now the people are more willing to accept change.

8 What Next?

The success of the PRISM programme has given Serco the momentum to continue with a new programme of work to implement ISO 20000:2005 IT Service Management[12]. PRISM 2 started in late 2007. Management have taken the decision to apply continuous improvement to the Service area for a number of reasons:

- To align IT services with the current and future needs of the business and its customers.
- To demonstrate to customers that a recognised certificate has been achieved for IT Service Management.
- To reduce the long term cost of service provision.

[12] ISO/IEC 20000-1:& 2 2005 Information Technology Service Management – Part 1 Specification & Part 2 Code of Practice

- To improve the quality of the IT services delivered.
- To provide a 'goal' that staff and Management can aim for, this will help to maintain momentum.
- Embed the culture of continuous improvement throughout the whole operation.
- Provide a standard approach, across all teams, regardless of type of work carried out. This will, over time, make day to day management easier.
- To establish core IT Service Management metrics.

The original organisation structure at Serco was not Service Management aligned and a recent decision was therefore made to implement a new structure. Although it is very early days this new structure is already starting to have an impact making everyone more aware of service priorities.

Overall the new project is adopting the same approach as the original Prism programme because of the success achieved. Why change a winning formula!

Finally – Be encouraged if we can do it, so can you!

9 Literature

M. E. Fagan, 'Design and code inspections to reduce errors in program development', *IBM System Journal,* (1976).

M. E, Fagan, 'Advances in software inspections', *IEEE Trans. Software Engineering,* (1986).

Office of Government Commerce (OGC) Publisher: Stationery Office 'Managing Successful Projects with PRINCE2 ™ Manual 2005' - 5th impression 2007

I.S. EN ISO 9001:2000 Quality Management Systems – Requirements

I.S. EN ISO 9004:2000 Quality Management Systems – Guidelines for performance improvements

ISO/IEC 20000-1:2005 Information Technology Service Management – Part 1 Specification

ISO/IEC 20000-2:2005 Information Technology Service Management – Part 2 Code of Practice

Bob McFeeley, Ideal^{SM}: A Users guide for Software Process Improvement, Feb 1996

W, Edwards Deming, The Deming Management Method, 1986

SEI, CMMI$^{®}$ for Development (CMMI-DEV), Version 1.2, 2006

10 Author CVs

Jill Pritchet, Software Measurement Services Ltd

Jill is a Chartered Information Systems Practitioner with over 20 years experience in the Software Industry. As a Consultant she uses her extensive experience to guide her clients to significant success in making change programmes really work.

Jill has a particular flair in the area of 'cultural change' her approach ensures that the necessary skills are fine tuned within the client organisation so that they build the in-house confidence to tackle any future change, (whatever framework is selected), and fully understand the issues they may face and how to overcome them.

As well as extensive knowledge of the various Software Process Improvement (SPI) methods, including ISO 9001:2000, ISO 20000:2005, ITIL and CMMI[13] **(SM)** Jill has given SPI tutorials and presentations at conferences across Europe encouraging others to learn from her experience including most recently ESEPG 2006.

Dom Michell, Serco

Dom Michell from Serco is an experienced software development and quality management professional with over 15 years experience. His career has been primarily in the Armed Forces developing mission and safety critical software for aircraft. For the last three years Dom has been the Compliance Manager at Serco with responsibilities for ensuring the quality of products and services and driving improvement programmes.

Dom has a particular interest in enabling service improvements by providing the correct environments and encouraging cultural changes to allow people to excel.

13 SEI, CMMI® for Development (CMMI-DEV), Version 1.2, 2006

Finding out what happened after SPI assistance in Ireland

Marty Sanders and Ita Richardson
Lero, The Irish Software Engineering Research Centre
University of Limerick

Abstract

Problems in software management have been addressed with solutions of various types and levels including software process improvement initiatives. The European Commission sponsored software process improvement (SPI) programmes in the 1990s to help make European countries more competitive in the global market. Several of these programmes took place in Ireland; one of the authors worked on all of these programmes and was in position to do research into any long-term benefits of the SPI programmes. The programmes used different standards and models (or none at all) as the basis for training and mentoring and no tool existed that would allow a fair assessment of these diverse environments. An interview guide and a model for assessment were developed to gather data about each company and this was analysed to evaluate the current state of software processes.

Keywords

Standards, assessment, MMAI, European Commission programmes, CMMI, ISO15504, ISO9001

1 Introduction

Software problems have existed since the earliest days [12, 16]. These problems have been met with many different solutions, such as improved processes and skills, and implementation of new methodologies and standards. In particular, the concept of software process improvement (SPI) has grown since originally proposed [5]. Several SPI standards and models now exist and have been used to help companies manage software development and maintenance more effectively. They also provide an opportunity for companies to show capabilities for products and services provision.

From 1994-1999 the European Commission sponsored SPI programmes. The research presented in this paper was designed to understand whether SPI assistance was useful in helping companies improve their software processes?' through:

- Evaluation of the commonalities and differences in selected standards and models
- Development of an assessment model that could be used in companies that used these different standards and processes for improvement
- Usage of the assessment model to evaluate the current status of these companies.

2 The software environment

To overcome problems in the software development environment, globally recognised standards are commonly used. According to Altman [1] 'International standards have become, at the same time, the price of admission to the global economy and the glue holding it together.' While he was not speaking directly about software, software process standards are increasingly being used within software companies. Commonly used standards initially were ISO:9001 1994 [6], used widely in Europe for quality management, and the Capability Maturity Model (CMM) [13], in the United States. ISO/IEC 15504 [15] was developed to define how to assess software processes. All three of these standards have evolved and changed since the nineties [3, 7]. Efforts at harmonisation have been made, but there are still differences in purpose, format and content [10, 18].

A detailed comparison of the CMM and ISO 9001 [14] and one of CMM and SPICE [15] shows both commonalities and the difficulties of matching up standards with one-to-one, one-to-many and other relationships, as shown in Table 1. It should be noted that such comparisons are frequently looking at general intent rather than specific wording. For example, Paulk [14] noted that '..a significant degree of judgment (and consequent possibilities of inconsistency) may be used when determining a reasonable relationship between the clauses in ISO 9001 and the practices in the CMM'. This also applies once SPICE is added to the comparisons.

Table 1 Harmonising CMM, ISO 9001 : 1994, ISO TR 15504 (SPICE)

CMM	SPICE	ISO 9001
RM		4.2, 4.3, 4.4.3, 4.4.6, 4.5.2, 4.19
PP	***CUS.2***	4.1.2.2, 4.2, 4.3, 4.4.2, 4.9.1, 4.9.2
PP	MAN.1	4.1.2.2, 4.2, 4.3, 4.4.2, 4.9.1, 4.9.2
PP	MAN .2	4.1.2.2, 4.2, 4.3, 4.4.2, 4.9.1, 4.9.2
PP	MAN.4	4.1.2.2, 4.2, 4.3, 4.4.2, 4.9.1, 4.9.2

3 Research environment

Between 1994 and 1999 the European Commission (E.C.) sponsored development of programmes to transfer knowledge on how best to implement SPI to help make European companies more competitive in a global marketplace. Two of the programmes were conducted only in Ireland, the rest throughout different European countries. These programmes were diverse and used different standards (CMM, ISO 9001, ISO TR 15504 [aka SPICE]), or nothing but good practice, leading to potentially very different types of software environments in the companies taking part. One of the authors worked on the development of two of these programmes and was involved in the execution of all. The research project required that we examined and compared heterogeneous software development environments.

3.1 Development of model for assessment

In the first instance, we examined other research and models which were used to gain information about software environments.

van Loon used a questionnaire for a three-week web-based survey, European-wide survey in the SPICE and knowledge management communities [19]. The questions were aimed at understanding each organisation's environment and how well it supports improvement and innovation. Appropriate questions about the organisation were selected from this survey for inclusion in the interview guide used in this research.

An early self-assessment questionnaire was used by the Software Engineering Institute (SEI) to gather data about the existence and maturity of specific software processes The first version (CMU/SEI-87-TR-23) was used as an appraisal method for assessing the software capability of contractors [22]. This covers only CMM processes so was not suitable for our project.

The Express Process Appraisal (EPA) uses six of the seven staged CMMI Level 2 processes (requirements management, configuration management, project planning, project monitoring and control, measurement and analysis, process and product quality assurance) [20, 21]. This is based only on the CMMI and again we deemed it not to be suitable for this research.

The Rapid Assessment for Process Improvement for software Development (RAPID) one-day assessments of 2000 were based on eight ISO/IEC 15504 processes (requirements elicitation, software development (assessed as ENG.1 in toto, not disaggregated), configuration management, quality assurance, problem resolution, project management, risk management, process establishment) and three capability levels (1=performed, 2=managed, 3=established). Over twenty companies responded and took part in the assessments, with a follow-on meeting held about a year later [2]. Specific questions were asked related to prioritised actions from the earlier SPI programme. This questionnaire is focused only on the previous RAPID assessments, based on ISO/IEC 15504 only, and is not suitable for this research.

In the case of the companies studied, we faced a heterogeneous environment – that of a diversity of companies who may have adopted a variety of software process management environments. None of the existing methodologies and models was suitable for this heterogeneity. Therefore, as a support to help us evaluate whether the implementation of the European Commission had a long term effect, we developed a model - Multi-Model Assessment Instrument (MMAI) and subsequently used this to help us understand whether SPI assistance was useful in helping companies improve their software processes.

3.2 Research methodology

Additionally, qualitative research methods were used. This included carrying out one-to-one interviews following a detailed interview guide. The purpose of the interview guide was to:

- Document the continuing history of an individual organisation in software process improvement, including the current environment and plans for the future;
- Enable cross-case analysis, both case-oriented and variable-oriented [11]

The ability of the interview guide to meet these requirements was verified by sending each interviewee the results of the interview and asking for any incorrect information to be corrected. Several concerns are accuracy, believability, comparability to other data of a similar nature and reportability, as well as how to remove any bias and prejudice [4]. Techniques to meet these concerns are to use: a reliable and valid survey instrument, checks to ensure internal and external validity and rules to minimise bias.

The data collected in this project was analysed in multiple ways, depending on the type of information collected. Part A of the interview guide covered information about the organisation. This was used generally to look for patterns which might be indicative of success or failure or just differences in implementation of SPI. Part B was used for assessment of specific software processes with numerical values given for the level of performance of defined activities. This shows more directly how well improvements were applied in the SPI assistance programmes.

In summary, this research can be described as positivism with an initial theory that a model can be developed and used to gain an understanding about the software environment of an organisation, a qualitative strategy of inquiry and narrative research to get stories of people and their organisation's histories. The method used was one-to-one interviews with several ways to understand responses, among them asking related questions in different ways or about different times. An assessment is used to provide details about specific practices currently being followed in the organisation. Cross-case analysis is used to look for commonalities and differences. Several factors are built into the survey to increase reliability and validity and reduce any bias.

Consequently, to answer the research questions, this research is a based on interviews which use both questions to get qualitative information about the organisation and its temporal experiences with SPI and an assessment tool to get more specific details about current software processes. It is a descriptive study to produce information on groups and phenomena that already exist [4].

The process followed is shown in Figure 1. Lists of companies who had taken part in the SPI assistance programmes were updated using web searches and access to other publicly available sources.

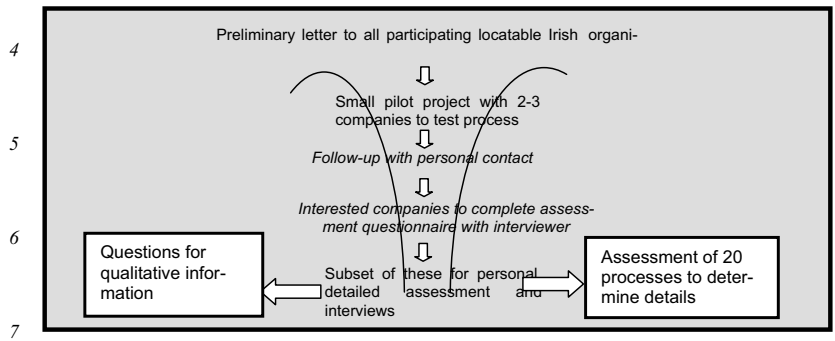


Figure 1 Planned research for gathering data on improvements

One interesting aspect in the development of a research population was the turnover of Irish software companies in the 9-14 years since the SPI programmes. Table 2 shows the responses of companies to phone calls, e-mails and cold calls at the door.

Table 2 Summary of response to request for interviews

No. Companies	Response
24	Do not know how to contact, number not in service
11	No answer, no response to calls or no return calls, phone constantly engaged
7	Phone disconnected or belongs to different company
6	No software development anymore
4	No longer in Ireland
2	New staff doing completely different work
totals 54	Total number of companies out of the picture
plus 23	Possible companies worth further contact
plus 3	Companies in Northern Ireland and excluded from survey
equals 80	Companies in SPI assistance programmes

Ten companies agreed to a full interview and assessment to validate the model for assessment. One small company agreed to a short interview but as they had not adopted the standard taught (ISO 9001 : 1994) a process assessment was not suitable.

The first part of the interview guide asks questions such as: main business and type of organisation; role of interviewee; size of the organisation and software groups and project teams; and project characteristics. One assumption is that if past activities, the current environment and plans for the future all show interest in and use of process improvement, the organisation shows a 'culture of improvement'. If this culture does exist, and if the process assessment supports the answers to the questionnaire, from this one can conclude the original SPI programmes are likely to have had some role to play in this culture, since for most companies this was their first exposure to the concept.

One of the concerns has been how to compare companies that followed different standards or none—what is a basic model of improved processes that could be applied to a company following ISO 9001 and another one that learned the CMM, for example? The option chosen to answer this question was to create a basic SPICE profile using SPICE as it was designed to be—a framework for other standards. This covered all the Level 2 and some of the higher-level CMM processes and most of the software-related ISO 9001 processes, making allowances for the different wording of the language.

The second part of the interview guide covers 20 processes common to ISO 9001 : 1994, the CMM and ISO 15504 (SPICE) Technical Report. These are the standards and methods that were used in some of the training. The model itself uses the SPICE processes, using SPICE as a framework for other standards, one of the original purposes for SPICE. Not all standards call similar actions by the same words; this research looked at activities with the same intent. One example is ISO 9001:1994 4.8 Product identification and traceability. This activity is called configuration management in other standards. Careful study of the wording to understand the intent was required to obtain consistency.

The processes selected for the assessment included components from all five SPICE processes categories. Customer-Supplier is used both for the company as acquirers of software or products used to create their own products and the company as supplier to its own customers. This was done to bring together ISO 9001, with its emphasis on controlling inputs and supplying quality products and the CMM, looking at correct understand of what goes into the software and subcontractor management under the SPICE more extensive umbrella.

Processes to include in engineering were software requirements analysis, design and testing. All support processes were included: documentation, configuration management, quality assurance, verification, validation, joint review, audit and problem resolution. Management activities include project, quality and risk management. Training and measurement were included as organisation processes.

This model can be compared to the standard for small software companies (ISO/IEC 29110) that is being developed by the ISO Working Group 24 [8]. This is currently in development but the processes included as of December 2007 are:

Project planning, *plan execution, assessment and control*, and closure
Software implementation initiation, *requirements analysis*, architectural and detailed *design*, construction, *verification, integration and test, qualification testing, validation* and product delivery
Supply, documentation, risk management, configuration management, quality assurance and review

The comments on the model are being processed in the summer of 2008 and a format or content change several times before final global agreement is reached.

The processes in common with the model used in this research are in italics; the words may be different since the words used in the standard are taken from another model. Because the assessment part of the interview guide developed for this research was based on components from three sources, the whole model is called the Multi-Model Assessment Instrument (MMAI).

4 SPI in Ireland, 1994-1999

Software Process Improvement assistance was provided to about 80 companies that took part in training and consulting programmes sponsored by the European Commission 1994-1998. The programmes had a variety of knowledge transfer methods - training only, consulting only and both training/consulting - as well as differences in standards and models used. Short-term benefits were discussed for each of these programmes in case studies and final reports. One of the comparisons of interest in this research was to see how well these short-term benefits have been maintained in the ensuing years.

5 Research Results

Before using the MMAI, it was verified by several assessment and standards experts, such as van Loon. The process was then used with two companies as an initial validation; no changes were made as a result. The MMAI gave a good understanding of the software produced by two organisations that had gone through large changes; one had been sold off, the other had sold off the main product line and started a new one. Both were maintaining ISO 9001 certification and had incorporated the processes from the SPI assistance programmes in their software management.

Data were collected from 10 companies on both the company's view about software - past improvement, current environment and future plans - and performance on specific processes. There was no attempt to look at anything above performance.

One rather surprising finding was that five of the companies assessed had 30 employees or fewer and five were larger, so this was not as much of a small company assessment as the researcher had expected. In spite of the findings by others that small companies are different from large companies [2, 17] many characteristics of the software environment evaluated in this research are the same for different sizes of organisations:

- Average project team sizes are usually six people or fewer.
- Development projects are usually six months in duration or shorter.
- Web-based distributed PC systems are the norm for most, either in current systems or planned migration from older centralised systems.
- Improvements were made in most companies, with size not appearing to be a factor. More importantly, most of these changes have been sustained in companies regardless of their size. Of the five companies with the highest average scores in process ratings, FD improved apparently without any influence from the SPI training, OS and SM are large companies, FM and TM are small companies; the last four show improvements related to the assistance programmes.
- However, the usefulness of training alone was not established. Every company that had a mentor visiting the company on a regular basis made improvements that were incorporated in the software process and used as the way to do business. No company that had training only could point to specific improvements that lasted.

- Standardisation of software processes, whether following a specific standard or not, is one of the greatest benefits recorded by companies of all sizes. In companies with high staff turnover, the standardisation appeared to play a very supportive role.
- ISO 9001 is current in four companies, SI and TM have 11-20 employees, FM has 21-30 employees, and JC has 51-100 employees. FD at 31-50 employees in the department assessed was ISO 9001 certified for several years until the decision was made their processes were far advanced beyond the standard. ISO 9001 certification has proved valuable in organisations of varying sizes including all but the smallest surveyed.
- Most companies of any size indicated good senior management support.
- The link between understanding of company vision and goals and the success of SPI does not appear to have any connection to size of the organisation.

The processes that show the most frequent improvements in literature are project management, configuration management and testing. Some processes from this research have been more successfully implemented than others. Of particular interest is the careful attention to the interfaces with customers and suppliers. Configuration management is the next highest and all report the use of tools and a stable process. Risk management and measurement scores are the lowest, but since risk management is often included as part of business management documentation and measurement can be part of tools used for other purposes, the scores may actually have been higher if a detailed assessment had been done.

6 Summary and conclusions

The MMAI was developed as part of this research project as a basic quality model which has two components:

(A) questions about the company, its past, present and future plans involving software management

(B) a more formal assessment instrument to evaluate the performance of processes common to the three most widely-used standards/models in software.

Based on the results of the interviews and assessments, the model fulfilled its purpose and is a significant contribution; it enabled the researcher in a relatively short time to get a rounded picture of the company and its progression through software management changes within the context of other events in the company. The pattern of related questions enabled the interviewer to go back and discuss previous answers if there seemed to be any inconsistencies. The inclusion of purpose and expected results in the assessment instrument enabled the interviewer to clearly express what information was needed from the interviewee.

The assessment instrument also confirmed the value of using SPICE as a framework in which to fit other standards. It is important to note this was not a SPICE assessment as defined in the standard; SPICE has more processes in it than would have been taught or used with these companies and there was no attempt to determine capability levels, but using a tailored SPICE profile in an assessment of basic process performance suited the needs of this research.

The MMAI was used in a diverse set of organisations and validated that it works as desired to evaluate the status of software management. No tool has been documented previously in the literature which is able to cover such a wide variety of basic processes in the most commonly used software standards in such a diverse set of organisations. Furthermore:

- The analysis showed the relative effectiveness of different types of assistance programmes with the companies interviewed; the value of mentoring was definitely established.
- Factors for success were identified, including the widely accepted management support, but other factors less widely discussed in literature.
- The possible benefits of ISO 9001 were identified, which might include regular visits by an auditor and a wide community using the standard.
- Future plans to take advantage of this and other research have been defined to create a potentially useful tool for small organisations in Ireland and elsewhere.

The original research question was to find out if there were long-term benefits in companies that received SPI assistance up to 14 years ago. In the case of those companies who took part in the research, all but two showed there were long-term benefits. Both of those two companies are highly process-oriented now but we cannot say it had anything to do with the SPI assistance. MM, the company that shows the lowest scores today and is the smallest assessed actually reported improvements at the time of training. It is now recovering from a major reduction in size which may account for the low scores.

In addition, the fact of so many commonalities between companies large and small is interesting. One conclusion that might be drawn is the business requirements for standards and process management is more important than size of the company in determining what improvement takes place and becomes incorporated in the software management process. However, that was really beyond the scope of this research.

The MMAI is independent of any single standard and can be used by a company looking for a subcontractor or partner, a national body wanting to determine the status of its country's software companies, or a researcher in a similar situation of looking at mixed environments. In a global marketplace, use of standards, or at least management of a core set of standard processes is becoming increasingly valuable for even small companies.

Acknowledgement
This research has been partially supported by the Science Foundation Ireland funded projects, Global Software Development in Small to Medium Sized Enterprises (GSD for SMEs) grant number 03/IN3/1408C within Lero - the Irish Software Engineering Research Centre (http://www.lero.ie).

Figures and tables

Literature

[1] Altman, D. (2006) 'You like tomato and I like tomahto', International Herald Tribune, 4 October, 14.
[2] Cater-Steel, A. and Rout, T. (2006) 'SPI program retrospective: evaluating long-term benefits for small firms', paper presented at *6th International SPICE Conference*, Luxembourg, 3-5 May
[3] Capability Maturity Model® Integration (CMMISM), Version 1.1, CMMISM for Software Engineering, (CMMI-SW, V1.1), Staged Representation (2002) CMU/SEI-2002-TR-029, ESC-TR-2002-029 [online], available at: http://www.sei.cmu .edu/cmmi/models/ sw-staged.doc [accessed 13 Sep 2007].
[4] Fink, A. (1995) *How to Design Surveys*, Thousand Oaks, CA: Sage Publications.
[5] Humphrey, W. (1990) *Managing the Software Process*, 1989 ed. reprinted with corrections made, New York: Addison-Wesley.
[6] *ISO 9001 : Quality systems, model for quality assurance in design/ development, production and installation* (1994) International Organisation for Standardization, Geneva, Switzerland.
[7] *ISO 9001 : 2000 Quality Management Systems – Requirements*, (2000) Committee draft, ISO/TC 176/SC 2/N 434 International Organisation for Standardisation, Geneva, Switzerland.
[8] *ISO/IEC PDTR 29110, WD2 Software Engineering – Lifecycle Profiles for Very Small Enterprises (VSE)* (2008[1]) JTC1/SC7/WG10 International Organisation for Standardization, Geneva, Switzerland.
[9] *ISO/IEC TR 15504 Software Process Assessment*, (1998) JTC1/SC7/WG10 International Organisation for Standardization, Geneva, Switzerland.
[10] Kitson, D. (2005) 'The CMMI® product suite and international Standards: an update' paper presented at the *4th International SPICE Conference on Process Assessment and Improvement*, Klagenfurt, Austria, 28-29 April.
[11] Miles, M. and Huberman, A. (1994) Qualitative Data Analysis 2nd ed, Thousand Oaks, CA: Sage Publications.
[12] O'Connell, F. (1994) *How to run successful projects II – the silver bullet*, Hemel Hempstead, Hertfordshire, UK: Prentice Hall.
[13] Paulk, M., Weber, C., Garcia, S., Chrissis, M. and Bush, M. (1993) *Key Practices of the Capability Maturity Model SEI-93-TR-025*, Pittsburgh, PA, Software Engineering Institute.
[14] Paulk, M. (1994) *A Comparison of ISO 9001 and the Capability Maturity Model for Software, CMU/SEI-94-TR-12; ESC-TR-94-12*, Pittsburgh, PA, Carnegie Mellon University, Software Engineering Institute.
[15] Paulk, M. (1999) 'Analysing the conceptual relationship between ISO/IEC 15504 (software process assessment) and Capability Maturity Model for software', presented at *1999 International Conference on Software Quality*, Cambridge MA.
[16] Pulford, K., Kuntzmann-Combelles, A. and Shirlaw, S. (1996) *A quantitative approach to Software Management: the ami Handbook*, Wokingham, England: Addison-Wesley.
[17] Richardson, I. and von Wangenheim, C. (2007) 'Why are small software organisations different?', *IEEE Software*, January/February, 18-22.
[18] Rout, T. and Tuffley, A. (2007) 'Harmonising ISO/IEC 15504 and CMMI', *Software Process Improvement and Practice*, Wiley 12, 367-371.
[19] van Loon (2005) 'Surveying the status of organisational culture, improvement and innovation in the SPICE and knowledge management communities', paper presented at the *4th International SPICE Conference on Process Assessment and Improvement*, Klagenfurt, Austria, 28-29 April.
[20] Wilkie, F., McFall, D. and McCaffery, F. (2005) 'An evaluation of CMMI process areas for small- to medium-sized software development organisations', *Software Process Improvement and Practice*, Wiley 10, 189-201.
[21] Wilkie, F., McCaffery, F., McFall, D., Lester, N. and Wilkinson, E. (2007) 'A low-overhead method for software process appraisal', *Software Process Improvement and Practice*, Wiley 12, 339-349.
[22] Zubrow, D., Hayes, W., Siegel, J. and Goldenson, D. (1994) 'Maturity questionnaire', *Special Report CMU/SEI-94-SR-7*, Pittsburgh PA, Carnegie Mellon University, Software Engineering Institute.

[1] Work in progress by JTC VSC 07/WG 24 to create a Technical Report that references other standards. Part 1 Overview; Part 2 Framework and taxonomy; Part 3 Assessment guide; Part 4 n Profile specifications; Part 5 n, Management and engineering guides, where n stands for each profile.

7 Author CVs

Dr. Marty Sanders, +353-67-24987, e-mail: martys@iol.ie

Dr. Marty Sanders has been involved in the software business for nearly 50 years ranging from developer, maintainer, manager, consultant and trainer through specialist in Software Process Improvement. She finally decided to get a PhD to take advantage of all that knowledge and experience. She has been active in development of international standards for many years and is currently working with WG 24 to develop the software standard specifically for small companies (ISO/IEC 29110).

Dr. Ita Richardson, + 353-61-202765, e-mail: ita.richardson@ul.ie

Dr. Ita Richardson is a Senior Lecturer, Department of Computer Science and Information Systems and Lero - Irish Software Engineering Research Centre. Her current research projects, are as Project Leader on Global Software Development for Small and Medium Sized Enterprises and for S-Cube EU FP7 Network of Excellence. Her specific interestes are in Software Process Improvement and its implementation in Industry, especially in small (< 50 employees) companies, as well as Global Software Development, focusing on Small Companies as part of the chain of development, particularly in the automotive and medical device domains.

Automated Acceptance Testing Using Fit

Geir K. Hanssen[1,2] , Børge Haugset[2]

[1] Norwegian University of Science and Technology, Department of Computer and Information Science, NO-7491 Trondheim, Norway
[2] SINTEF ICT, NO-7465 Trondheim, Norway
{ghanssen,hau}@sintef.no

Abstract. Automated acceptance testing is a new and promising agile testing approach. Fit is the most established technical framework for specifying and executing acceptance tests which, ideally, lets the users express requirements in the form of acceptance tests. We performed an industrial case study to learn more on the costs and benefits of Fit tests. We learned that Fit tests may improve important parts of an agile development process but there is still a need for further research and improvements.

Keywords: Automated Acceptance Testing, Agile, Fit

1 Introduction

Testing in agile software development [1] is a fundamental practice to enable visibility of progress and to enhance communication and feedback within the development team and ultimately with the customer. Agile development is a change-driven process and it is vital that new and changing requirements are backed up with tests that can be defined quickly to have a constant view of the state of the evolving software product. So far unit testing and test-driven development are the most known, practiced and researched approaches to agile testing [2, 3]. However these are testing approaches that focus on the functional and technical level. These tests are defined by developers - for developers and give no direct value to customers or users which are central stakeholders in any agile development project. Over the past few years a complimentary testing approach in agile development has emerged – automated acceptance testing (AAT). In principle, the customer or his representative is given the role of expressing requirements as input to the software team paired with some expected result – an acceptance test. Whilst unit testing is about testing low level units such as methods or equivalent, acceptance tests integrate at a higher level, testing business logic. In this way a customer or potential user can relate to tests and it can be an efficient way of communicating precise requirements from customer to developers. Doing acceptance testing manually will in most cases be tedious, expensive and time consuming, and does not fit the paradigm of agile development which relies on instant feedback and short development cycles. Automation of acceptance tests may thus seem as a promising initiative to ease and speed up this process. The basic idea of AAT is to document requirements and desired outcome in a format that can be

automatically and repeatedly tested – very much based on the same philosophy as for unit testing, but on a higher level, understandable to users. One of the most prominent solutions for AAT in agile development is the Framework for integrated tests (Fit) [4] which provides a simple mechanism for documenting requirements as acceptance tests. Tests work the code through fixtures that integrate with business logic and can be used for regressive testing of the growing software product.

Fit is supposed to solve three main problems in software development [4]: 1) Communication: Ordinary written requirements can be ambiguous, prone to misunderstandings and lead to wrong design - Fit tests tend to improve the communication of what a software product should do through concrete tests that are based on customer-defined examples from the business domain. 2) Agility: Fit tests are aimed at supporting frequent changes; tests ensure that any changes in the software do not break previously satisfied requirements – thus enabling agility. 3) Balance: Fit tests are aimed at helping a development team to spend less time on gaining balance with fixing problems by reducing the number and severity of problems, catching them early, and making sure they don't return.

The aim of this paper is to address these potential high-level benefits as well as more detailed aspects through an industrial case study. Can we – from practice – see these effects? We have interviewed four professional developers from a Scrum development project applying Fit tests to learn from their experiences. We build this work on a thorough overview of similar experience reports on the use of AAT and Fit. We continue this paper by first describing Fit in section 2, followed up by a brief literature review on the current state of research on Fit and automated acceptance testing in section 3. We then describe our research method for the case study - see section 4. In section 5 we report the results from the case study. In section 6 we discuss our findings and relate them to the intentions of Fit. section 7 summarizes and concludes our work and points to future research.

2 Automated acceptance testing using Fit

Fit test descriptions are organized in tables. A simple example can be seen in **Error! Reference source not found.**. This test describes a system doing division on numbers. When running the tests, the input 'numerator' divided with another input 'denominator' should give the result written in 'quotient?'

Fit isn't automatically connecting these tables to the underlying business code it is verifying. This connection is made by using fixtures, which are different ways of reading the tables and translating them to calls in the business code. Several fixture types are included in the tool, but they can also be made from scratch. Three types of tests are:

- ColumnFixture, for testing calculations: Testing according to business rules. Certain input values shall result in certain output values. **Error! Reference source not found.** is an example of this type.

- ActionFixture for testing actions: The test defines a series of actions such as acting out calls for buttons pressed, values entered into input fields, etc. The running test reports if the application gives the appropriate results. (actions are start, enter, press, and check)
- RowFixture for testing lists and other collections: A set of input or actions that constitute a single group and need to be tested together. The list of elements does not necessarily need to return correct values in any particular order.

The Fit tests can be run at any moment after they are created, even if the underlying code hasn't yet been created. The result of running a Fit test varies accordingly, see **Error! Reference source not found.**. A green result means the test has returned the desired value, and has passed. A red test result means the test failed, while yellow means part of the test or something else went wrong. A grey test result shows that the test hasn't been processed. Here we see that the division code returns correct values for the two first pairs of input values, as the output value fields are coloured green. The third division result is red, indicating a test failure. The expected result is shown together with the actual result from the call. The depicted example is a very simple one. The actual Fit tests in a working condition could be much larger. It shows, however, that the direct system requirements of the wanted procedure are possible to write in a table format, hopefully easy to understand from a users viewpoint in contrast to unit tests which normally only is meaningful to developers.

Division		
numerator	denominator	quotient()
10	2	5
12.6	3	4.2
100	4	25 *expected* 22 *actual*

Fig 1 Fit test table execution

Since the Fit tests themselves are made in a simple table format, many tools can be used to create them. This includes Microsoft Excel, HTML and a wiki. FitNesse is a software testing tool, a wiki and a web server combined. Since it is a wiki, adding and maintaining tests is easy. It is based on the Fit framework. FitNesse is available for free at www.fitnesse.org. While FitNesse is written for Java, versions for other programming languages such as C++, Python, Ruby, Delphi and C# exist. Note that the developers of FitNesse express that FitNesse should be used in addition to unit tests, and not instead of them[1].

3 State of the research

Fit seems to be an easy tool to learn. The average introduction of the basics is a 3-4 hour crash course plus relevant readings. Melnik et al. report that 90% of technology students managed to deliver their Fit test assignments after a brief introduction [5]. Unfortunately there isn't much clear evidence about non-technical users. Business

[1] http://fitnesse.org/FitNesse.AcceptanceTests

graduates (acting as clients/users) found it hard to learn, but eventually produced good specifications with ease [6]. The teams mixed of business and computer graduates were no better at describing a good quality specification than pure computer graduate teams. One company describes successfully adopting and using Fit tests. However, this specific customer had an information systems background [7]. When asked whether they would use Fit again, two studies show positive responses [6, 8]. Findings show that there was no correlation between students' own perception of the value of the tool and the quality of the requirements specification they produced [6].

The general impression is that Fit is a good tool for specifying requirements. Melnik and Maurer found that 80% of students in an experiment preferred requirements specified in Fit as opposed to prose [9]. Similarly, Read et al. noted that the average student in his experiment felt Fit to be adequately appropriate for defining functional requirements [8]. The same students rated writing Fit tests as just as difficult as writing JUnit tests and requirements specifications in prose. 43% of them found the Fit specifications good enough to implement a web service. 26% managed but had problems. In a small survey of test notation tools by Geras et al., Fit scored best on 'ease of use' (together with scripts and XML) [10].

Interestingly, in an industrial case study Prashant et al. found that customers don't always want to write the tests [11]. The authors suggest, in hindsight, that the Fit documents should rather be described as specifications. Likewise, Melnik and Maurer mention the use of previously-made info-sheets (diagrams, mock-ups and call-outs) that the customer brings to each iteration meeting [7]. This made it easier for the customer to explain the context to the developers. In this particular project the customer ended up writing 40% of the tests, the developers and testers wrote 30% each. The tabular structure of the Fit tests was seen as superior to XML for writing tests [12]. Being able to write test code needs to be easier than writing normal code [13].

Writing acceptance tests has at times been given lower priority, because of the complexity [7]. While finding the time (or inspiration) to write Fit tests might at times prove hard, there are some claims that doing just so leads to a better specification. Fit has been attributed to help discovering a lot of missing pieces and inconsistencies in a story [Ibid]. This is potentially related to effects described by Melnik et al., where they claim that Fit reduces noise, over-specification, and ambiguity [5].

One effect when projects grow large is the growing need to organize the specifications. Prashant et al. report that one company started storing tests in different directories according to their lifecycle status [11]. They also think their successful refactoring of a large application part was due to their extensive readable Fit test coverage. The use of semi-automated Fit test creators has been suggested to help with these problems [14].

Melnik et al. found no evidence to support that good quality acceptance test specifications made by a customer team resulted in better quality implementation by a developer team [6]. They think this is an issue which requires further study.

One problem in using tools for development lies in letting the tool drive the collaboration and not vice versa. Interestingly, Prashant et al. noted in their study that one group claimed Fit initially hindered communication [11]. They focused too much on preparing 'syntactically correct Fit documents' for the development team, and didn't focus on developing the most appropriate specification. 41% of student groups

developing software using Fit had no contact with the test suite development team during development [8]. The Fit suite summary has by some student groups been used as a project dashboard [15], this could also be used by project customers as a feature to keep track of project status.
According to Prashant et al., the Fit specifications should be the main definition of the specification, and should be integrated in the continuous build process [11]. Abstraction should not come in the way of readability. Organizing a team is always a struggle. One team ended up with so many Fit documents that they created a special team to deal only with this [11]. This increased the Fit development, and enabled analysts to effectively write detailed Fit documents. The result was that developers ended up writing code to make the tests pass rather than cooperating with the original customer. The company increased staffing from 6 to 24 developers in four weeks midways in a project, yet still succeeded. This they claimed to be partly because the Fit documents were readable, living documents that new developers could use to understand the domain. The large Fit test coverage also gave them the impression that unanticipated changes that introduced errors would be picked up by the system. The authors however call for caution about adding more developers without adding more analysts. For them, the same amount of analysts spent less time collaborating with developers, and more time specifying Fit documents. This resulted in more defects getting through to the application.
Coding to make an acceptance test pass can take a lot more time than for unit tests [16]. The problem arises when trying to separate between 'unimplemented failures' and (the more important) regression failures. A possible solution made by Deng et al. was to create an Eclipse add-on, FitClipse, which remembers if a test has previously passed or not. Another problem lies in the assumption that a passing test suite means the code is good. Melnik and Maurer touch briefly on the topic when they describe the "deceptive sense of security" students had when the tests passed, and warn about not thinking outside the box [9]. Fit got the lowest score (2/12 points) of all test notation tools surveyed (albeit not a completed survey) when used for describing non-functional requirements [10]. This included performance, security, and usability.

4 Case study method and context

To build on the knowledge base on Fit and AAT we studied a software development project at a medium sized Norwegian consultancy house that applied both Scrum and the Fit acceptance testing framework. The case project is an extension of a previous delivery starting in 2003 and was still ongoing at the time of the study. The team varies from 5 to 10 persons. The customer consecutively pays for resources spent (time and materials). The system being maintained is part of a larger system of subsystems delivered by other suppliers. The development team adopted Fit about a year ago. Fit tests were mainly used to test interfaces to external components. To collect data from the case project we interviewed four developers about their experience in using Fit tests. Each interview was made on-site lasting for 30-40 minutes and was done semi-structured (using an interview guide) to allow the respondents to reflect. The interviews were recorded and transcribed by a professional

language service company. The resulting four interviews were analyzed according to the principles of constant comparison [17], meaning that we have manually searched the transcripts to discover relevant experience and effects of AAT. In the analysis we followed the structure of the interview guide to identify conformance or divergence in the respondents replies. The following section presents a summary of the interviews, as it would be too voluminous to reproduce the whole text (approximately a 100 pages of text).

5 Results

Results are reported according to the three main sections of the interview guide; 1) how automated acceptance testing was used, 2) effects and results and 3) experience – looking into both positive and negative aspects.

5.1 About the use

Who writes the tests? Tests were written by the developers, no person had a dedicated responsibility for writing the acceptance tests and most noteworthy – no customers participated in expressing tests. The developers had worked so long in the area that they felt they at times had better knowledge of the field than the customers themselves. This has to be seen in relation to the team also using Fit tests mainly to test interfaces at a system services level. Each developer decided if tests were needed and flowingly had the responsibility of writing tests for the feature he/or she worked on. Sometimes this was based on consulting others in the team. The project appointed a person in charge of running the tests as well as reviewing them.

How were tests written? In the very start of using acceptance tests developers tended to define tests for most issues, also including simple ones. This has eventually changed over time, now only the most complex features are covered by acceptance tests – simple issues are considered to be not worthwhile to test this way.

When were tests written? Tests were written when the need occurred, meaning that they did not define any specific phase or point of time in the iterations to do so. As the system under development grew the need and motivation to define acceptance tests also grew.

Training, support and introduction The introduction of this new practice was made as simple as possible; one expert external to the project gave a one-day introduction in the form of a workshop to the developers.

Positive and negative tests In a few cases negative tests were written, meaning that a test is meant to provoke an error; this is used to ensure that the system handles errors

as intended. One developer commented that it is hard enough to develop and maintain positive tests.

Maintaining tests / keeping tests and code in sync Besides development of code, the tests were also modified either as an outcome of test results or that a customer reported an error. The developers found that web-services were stable (to change) and thus the need to update these tests was low. In general there seemed to be a difference in code/test stability of core components or services that are stable and need less updates of tests, while code and tests closer to the GUI tends to be more unstable and cost more in terms of maintenance.

5.2 About the effects, outcome and results

How use of AAT affects communication and cooperation with others (team and outside) Most developers reported clearly that having a part of the code covered by acceptance tests made it safer and thus more convenient to let others work on their code and vice versa. Similarly the acceptance tests increased clarity in the sense that all developers could see what the code is intended to do and the current status. Tests were easier to grasp than the respective code, and acted as spreaders of competence. Also – in some cases where a developer sees that she will have to alter another developer's code in such a way that the test will break she gets an initiative to contact the original developer and discuss the issue. Except from rare cases, the customer does neither specify tests nor evaluate the results – they perform their own testing.

Tests improving understanding of the system The developers expressed strongly that writing acceptance tests, preferably upfront, improved their understanding of the domain and the system under development, however it did not improve the understanding of the code itself. One developer also compared it to unit-testing and underlined that the clear separation of test and code in acceptance tests enabled him to get a better overview of what the system is intended to do as in contrast to how it does it, technically. Another developer commented that these effects are reduced when tests grow large.

Requirements management As the acceptance tests were not used for communicating with the customers they were neither used for documentation of requirements.

Process control The use of acceptance tests seems to contribute to the visibility and process control of Scrum. Test results show graphically which modules that are finished.

Customer satisfaction We asked about the developer's subjective opinion on how the use of acceptance tests affects customer satisfaction. The general impression is that these tests help the developers to identify and correct more errors, and that fewer errors consequently give better customer satisfaction.

Product quality The developers share the opinion that their use of acceptance tests positively affects product quality in two related ways; first more errors are handled and secondly regressive testing frees time compared to manual testing. However, testing was usually not prioritized in extremely busy periods of the project.

Documentation The growing suite of acceptance tests acts as a good extra documentation of the system, yet on the behavioral level – such tests do, by concept, not indicate how functionality is implemented. The tests were not currently being used as documentation for the customer, but some developers stated that this would perhaps be something for later. The customer had for historical reasons not paid much attention to documentation.

Fixture/test fatness/complexity Fixtures and tests grew large, typically resulting in many columns in Fit. They can be difficult to follow and thus sometimes large tests are broken down into several simpler ones. In general the developers try to keep the tests simple in the first place.

5.3. Experience

What does it solve? Each respondent was asked to summarize their positive experiences from the use of automated acceptance testing, both directly to themselves and for the total project. Top-rated issues were:

- It is safer to make changes in code
- We get fewer errors
- It is easier to share competence within the team due to perceived increased safety (of altering code)
- It reduces the need to do manual testing, which is a time-saver
- Compared to unit-tests, the acceptance tests give a better overview. It is also easier to add new tests
- Writing acceptance tests makes you think of what you are going to make before you do it. It also makes you find special cases that may be extra important to test properly
- Usually specifications from customers are poor so writing acceptance tests is in a way reflection on requirements
- It is very comforting for the developers to know that the system (or relevant parts of it) works
- In cases where external systems are unavailable at development time, acceptance tests acts as a good substitute.

Frequency of finding errors/issues Most errors are found shortly after the introduction of the case when the code is under development. However, change of test data in e.g. a database may also cause a test to fail, even if the code is unaltered. We see that tests themselves acts as documentation for the developers preventing later changes from breaking the tests.

Discovery of requirements that otherwise would have been missed Broken tests can also indicate disparities in the version control system, typically if someone has forgotten to update changes - tests may unveil not only code issues but integration issues as well. In general, acceptance tests may also help to reveal complex issues that by nature are unsuitable to time consuming manual testing.

Main negative experiences and problems When asked to summarize negative aspects of AAT the following issues were reported:

- Having a test is by itself not a guarantee; it can be easy to fool oneself by writing poor tests
- One obvious cost of adopting any testing practice, including AAT, is the time used to define and maintain tests in synchronization with the code being tested. Particularly maintenance of tests is reported to require a lot of effort. This gets even more prominent when tests have hard-coded data for check of results
- Typically, when a developer is new to this testing strategy tests can easily grow very large – it takes experience to establish a proper test design
- Some developers found it to be hard to write proper tests due to many awkward rules in the domain and the business logic of the system being developed. Such tests easily get complicated with the effect that they are both hard to understand and modify.

Is it relevant to continue to use AAT, any modifications of practice? All the developers responded positively when asked if they would like to continue to use AAT in new projects. They underlined that it would be a great advantage to start using the practice from the very start of the project. Some developers also commented on the need to be deliberate on what to test, and to test small parts of the system to avoid large tests that are cumbersome to handle.

6. Discussion

We started out by referring to some intended effects of using Fit tests. Can we find any support for these from our and others studies?

About communication: We see clearly from our case that Fit-tests were not used for communication between users and developers. Development team members wrote these tests themselves based on their experience with the application domain and of course based on other communication with the customer side. Other studies are inconclusive on this issue. Prashant et al. [11] commented that customers may not want to express tests in this way. Melnik and Maurer point out that extra means of communication (info-sheets) are established [7]. This may indicate that expressing requirements in the Fit table format still is too restrictive. This is aligned with the media richness theory [18] that explains an increase in communication efficiency with an increase in media richness. Another aspect of communication is the internal team communication. Here we see from both our case study and other relevant studies that

having features of the software product covered by tests leads to higher confidence in the development team. As the code is ”protected” by tests and thus safer to alter, developers claim that is also easier to share competence in the team.

About agility: Using Fit tests is supposed to support design changes when the needs of the business change. In our case, at the start developers tended to match all features with proper Fit tests. Demanding on resources, the practice changed to only covering the more complex features with tests. Tests were written when the need occurred, as defined by the developers. In this sense writing Fit tests seems to enhance agility – over time the test suite grew and formed a safety net that affected how flexible the developers could be with respect to quickly alter design and code. Developers would instantly see if changes negatively affected already completed features.

About balance: Using Fit is supposed to reduce time to gain balance by quicker recovery from problems. We got clear feedback from the interviewed developers that their use of Fit helped them find more errors in less time than with manual acceptance testing. This freed up time to spend on developing new features.

Further on, our findings show that very short and simple courses seem sufficient to learn the basics of Fit – also for non-technical customers. Even so, we found that it took some time and experience to be able to see which parts of the system that need test coverage – aiming for 100% test coverage might not be relevant. The project we studied did not engage customers in writing tests, however Read et al. [8] indicate that persons without an IT background may find it harder to grasp. We find no industrial experience on what it takes for non IT professionals, such as most customers are, to specify requirements using acceptance tests. Yet it is likely that introducing AAT to customers would require more extensive training and support. We know of only one study report that discusses quality of AAT in relation to quality of implementation. Melnik et al. found that there was no such correlation [6] and state that further research is needed. Writing acceptance tests may make developers reflect on the design and system behavior in advance of programming. A similar effect, general to up-front test definition, can be seen from studies of test-driven development [2, 3]. Developers also reported the tests to be valuable documentation of the intended behavior of the system, thus resembling unit tests but on a higher level.

One of the obvious reasons for any type of testing is to discover errors, non-conformance or flaws that need to be resolved. We see from Reads student experiment that developers discovered both inserted errors and even non-intentional errors [8]. Our interview respondents reported that the new testing practice helped them find and remove more errors than before. They also felt it made them able to see integration issues more clearly. One of the most evident findings in the interviews was that developers found it more convenient to share code with other developers. Hence we see that the use of acceptance tests may improve communication and cooperation in the development team. This is in contrast to the experience in one of the other studies. Prashant et al. observed that using acceptance tests (in the early phases) actually hindered communication as developers put too much effort into expressing correct Fit documents, on the expense of expressing only appropriate tests

[11]. This seems to be a balancing act; benefit should be weighted against cost. This balance deserves further research.

Only a few studies addresses the customer communication aspect of AAT: Melnik and Maurer found that engaging customers with an information systems background resulted in a successful adoption of Fit [7]. As a contrast, Prashant et al. observed, like us, that customers preferred old-fashioned requirements descriptions in plain text [11]; so this matter is not fully investigated. While AAT shows to be beneficial we also find issues related to costs. Naturally, defining tests costs time and attention. Having established a test that is closely connected to the code also implies a need of maintaining the test to keep it synchronized with the code. Deng also underlines this [16]. The respondents we interviewed reported maintenance to be a considerable cost, especially when the tests also contained hard-coded data for checking results. Another important issue found, that applies to testing in general, is that it is easy to write a test that completes successfully, yet it may still not cover the code properly. Thus, writing tests requires careful consideration or else they can give a false sense of security. None of the studies forming the base for this discussion report quantitative data documenting the net benefit of AAT. We here rely on developers' preference of continuing to use AAT based on their experience as an indication of the feasibility of this testing practice. All of our respondents replied positively when asked about this, some even commented that it was fun. Other studies also report this opinion amongst developers [6, 8], similarly we would also like to see studies of customer opinions.

7. Conclusions and further work

Automated acceptance testing surely holds a great promise. The limited experience base tells us that some important gains have been achieved. Yet there is still a great need for investigating this testing practice further. We have pointed out some new topics and concepts that deserve closer investigation in later studies. Especially we would like to pursue the role of these types of tests for communicating better with the customers, something we will aim for in later studies. We would also like to see more quantitative studies investigating process and product effects such as process control and product quality. Finally, we would like to sincerely thank the case company for giving us access to do interviews. This work was partly funded by the Research Council of Norway.

8. Acknowledgements

We would like to sincerely thank the case company for giving us access to interview their developers. This work was partly funded by the Research Council of Norway under grant #179851/I40.

9. References

1. Dybå, T. and Dingsøyr, T., *Empirical Studies of Agile Software Development: A Systematic Review.* Information and Software Technology 2008. **IN PRESS**.
2. Erdogmus, H. and Morisio, M., *On the Effectiveness of the Test-First Approach to Programming.* IEEE Transactions on Software Engineering, 2005(31): p. 3.
3. Müller, M. and Hagner, O., *Experiment about test-first programming.* Software, IEE Proceedings, 2002. **149**(5): p. 131-136.
4. Mugridge, R. and Cunningham, W., *Fit for Developing Software: Framework for Integrated Tests (Robert C. Martin)*. 2005: Prentice Hall PTR Upper Saddle River, NJ, USA.
5. Melnik, G. et al., *Suitability of FIT User Acceptance Tests for Specifying Functional Requirements: Developer Perspective*. 2004, Springer. p. 60–72.
6. Melnik, G. et al., *Executable acceptance tests for communicating business requirements: customer perspective*, in proccedings of *Agile Conference, 2006*. 2006
7. Melnik, G. and Maurer, F., *Multiple Perspectives on Executable Acceptance Test-Driven Development*. 2007, SPRINGER-VERLAG. p. 245.
8. Read, K. et al., *Student experiences with executable acceptance testing*, in proccedings of *Agile Conference, 2005. Proceedings*. 2005
9. Melnik, G. and Maurer, F., *The practice of specifying requirements using executable acceptance tests in computer science courses*, in *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*. 2005, ACM Press: San Diego, CA, USA.
10. Geras, A. et al., *A Survey of Test Notations and Tools for Customer Testing*, in proccedings of *Extreme Programming and Agile Processes in Software Engineering*. 2005
11. Prashant, G. et al., *Creating a living specification using FIT documents*, in proccedings of *Agile Conference, 2005. Proceedings*. 2005
12. Mugridge, R. and Tempero, E., *Retrofitting an acceptance test framework for clarity*, in proccedings of *Agile Development Conference, 2003. ADC 2003. Proceedings of the*. 2003
13. Andrea, J., *Envisioning the Next-Generation of Functional Testing Tools.* Software, IEEE, 2007. **24**(3): p. 58-66.
14. Amyot, D. et al., *UCM-driven testing of web applications*, in *Sdl 2005: Model Driven, Proceedings*. 2005. p. 247-264.
15. Read, K. et al., *Examining usage patterns of the FIT acceptance testing framework*, in *Extreme Programming and Agile Processes in Software Engineering, Proceedings*. 2005. p. 127-136.
16. Deng, C. et al., *FitClipse: A Fit-Based Eclipse Plug-In for Executable Acceptance Test Driven Development*, in *XP2007*. 2007.
17. Seaman, C.B., *Qualitative methods in empirical studies in software engineering.* IEEE Transactions on Software Engineering, 1999. **25**(4): p. 557-572.
18. Daft, R.L. and Lengel, R.H., *Organizational Information Requirements, Media Richness and Structural Design.* Management Science, 1986. **32**(5): p. 17.

TestPAI: A proposal for a testing process area integrated with CMMI

Ana Sanz, Javier Saldaña, Javier García, Domingo Gaitero
Computer Science Department
Carlos III University of Madrid
Avda. de la Universidad, 30, 28911 Leganés (Madrid)
asanz@inf.uc3m.es, jsaldana@inf.uc3m.es, jgarciag@inf.uc3m.es, domingo.gaitero@atosorigin.com

Abstract

Today, there are different organizations which are using CMMI to improve their processes. In many cases, they are interested in improving their testing process in parallel with the improvement of other processes. But CMMI does not provide the necessary support to improve testing process. In this paper, we analyze different testing reference models and propose a new process area to develop testing process improvement. TestPAI is a detailed defined process area which is integrated with CMMI to improve testing process. TestPAI has a structure similar to CMMI and includes all practices related to testing. Moreover, we have implemented TestPAI in a real organization successfully. For this, we defined a new testing process and modified other existing ones. Finally, we evaluated this new process to check its viability.

Keywords

Software Testing Process, Testing Process Improvement, Testing Reference Model, CMMI, TMM, TMMi, TPI, TMap

1 Introduction

The quality of a software system is mainly determined by the quality of the software process that produced it [1]. Likewise, the quality and effectiveness of software testing are mainly determined by the quality of the testing process used. So, a well-defined, managed and controlled testing process that is developed in parallel with the software development process will increase the quality of software products.

The most important benefits that software organizations obtain from testing process improvement are [4]:

- An increase in client satisfaction through the use of software products with lower defect rates.
- An increase efficiency in the software development processes.
- Facilitation of definition and fulfillment of quality objectives.

A well-defined process involves identifying and establishing test goals, test policies and test strategy; establishing and maintaining the test plan; defining specifications to create suitable test cases; analyzing and reporting the results obtained during the test execution and identifying and establishing suitable actions to solve the problems found during the testing process.

At present, there are different organizations which are working with CMMI to improve their processes. According to the world-wide maturity profile elaborated by the Software Engineering Institute (SEI) [2],

there are about 2.140 such organizations. Many are interested in improving their testing process, but CMMI does not provide the necessary support and the integration among CMMI and other reference models is very costly, which is an important problem for organizations.

The hypothesis of this work is to develop a testing process area which contains all practices related to test and which is integrated with CMMI. In this way, improvement in testing process will be done at the same time as improvement in other engineering processes, whenever CMMI is being used as the improvement model.

The main goals of this work are:

1. Define a testing process area to improve the testing process.

2. Implement the testing process area in a real organization.

3. Establish and apply a method for self-evaluation of a testing process based on SCAMPI[3].

This paper is structured as follows. Section 2 describes works related to testing process improvement. Section 3 specifies the testing process area: TestPAI, which had been developed and section 4 describes the implementation of TestPAI in a real Spanish organization to validate it. Finally, section 5 contains the conclusions extracted from this work.

2 Description of Testing Reference Models to Improve Testing Process

A reference model aimed at the testing process defines the necessary reference frame to determine the strengths and weaknesses of the testing process implemented in the organization. In other words, to determinate the current state of the testing process and to establish the improvement plan, it is necessary to collect the available practices and compare them with a reference model.

CMMI [5] [6] is the most widespread reference model in the software industry. So, if we want to improve the testing process at the same time as we improve other processes, we need a testing reference model which is compatible and integratable with CMMI.

Currently, the most important testing reference models are: TMM [7], TMMi [8][9], TPI [10] and TMap [11][12]. However, CMMI will also be analysed because it is important to know if CMMI is a suitable model to improve a testing process.

To analyze these reference models, we had defined six criteria. Table 1 shows and explains them.

Criteria	Definition
Type	Type of improvement model. It can be a process improvement model if it covers different process areas or a testing process improvement model if it only covers testing process area.
Fully covered	To be *fully covered* every practice related to test is included in the reference model.
Fully defined	To be *fully defined* every practice, which is included in the reference model, is completely detailed.
Staged representation	To be *staged representation* the reference model must allow to establish the organization maturity level.
Continuous representation	To be *continuos representation* the reference model must allow to establish the process area capability level.
Compliance with CMMI	To be *compliance with CMMI* the reference model structure must be compatible with CMMI structure and the integration between the reference model and CMMI must be possible in a easy way.

Table 1: Defined criteria to analyze testing reference models.

Table 2 summarizes the obtained result of the testing reference model study that we had made considering the previous criteria.

Criteria	CMMI	TMM	TMMi	TPI	TMap
Type	Process improvement model	Testing process improvement model	Testing process improvement model	Testing process improvement model	Methodology for testing improvement
Fully covered	No	Yes	Yes	Yes	Yes
Fully defined	Yes	Yes	No	Yes	Yes
Staged representation	Yes	Yes	Yes	No	No
Continuous representation	Yes	No	No	Yes	No
Compliance with CMMI	Yes	No	No	No	No

Table 2: Testing improvement models.

After these models were analyzed, we determined that none of them solved the problem described previously: to improve a testing process at the same time as the other engineering processes being improved. So, in the next section we propose a new process area for test improvement.

3 TestPAI – A testing process area integrated with CMMI

TestPAI is a testing process area integrated with CMMI. It is placed in level 3 with engineering processes. TestPAI was developed to provide the suitable reference frame to improve the testing process in parallel with the improvement of other processes implemented in the organization.

The main purpose of testing is to detect errors and check the right behavior of every individual component and the complete system. There are more and more small and medium organizations that are conscious of the importance of testing process improvement. Most of them need a framework which provides them the necessary tools to carry out the testing process improvement in an easy and not expensive way. Also, many of these organizations are working with CMMI to improve their processes; therefore they need a framework which must be compatible and integratable with CMMI. So, TestPAI emerged to provide it and solve that problem.

Before TestPAI was created, the authors had studied other possibilities and existing models. They elaborated a study of different reference models which could be the solution to this problem (see Table 2). However, none of them satisfied completely the necessities which must have a testing reference model. So, they decided to develop a new one: TestPAI.

TestPAI includes and defines all practices related to testing. It has the same structure of CMMI and is defined for staged and continuous representation. This permits the complete integration between TestPAI and CMMI.

TestPAI proposes five specific goals (SG) and their corresponding specific practices (SP). Figure 4 shows these goals and a brief description of each.

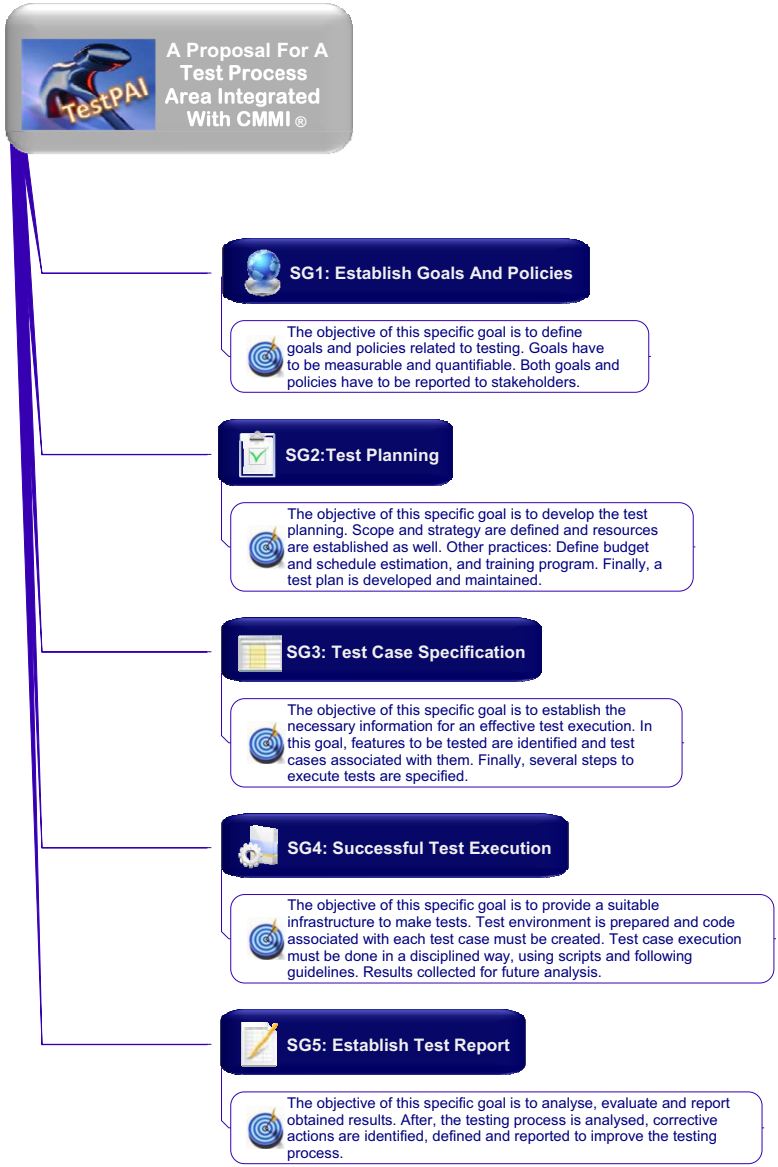


Figure 1: Specific Goals of TestPAI.

Specific goal one has associated two specific practices. Goals are defined to know what has to be done and policies are defined to show the guidelines to achieve these goals. Figure 5 shows this specific goal.

Figure 2: Specific Practices of SG 1.

Specific goal two has associated six specific practices. Making a test plan involves establishing and analyzing different items which affect testing management and development such as: scope, risk, resources, etc. Figure 6 shows this specific goal.

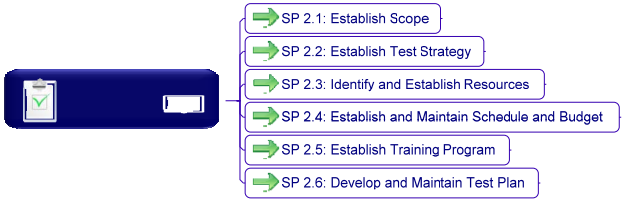


Figure 3: Specific Practices of SG 2.

Specific goal three has associated three specific practices. A well-defined testing process involves defining features that are not described in the test plan such as required inputs, desired outputs or the procedure to be followed. Figure 7 shows this specific goal.

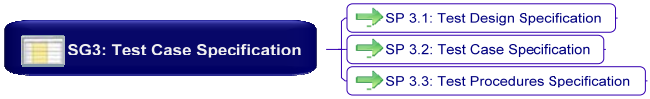


Figure 4: Specific Practices of SG 3.

Specific goal four has associated three specific practices. A successful test execution is an important item for improving software quality and preparing test infrastructure correctly is also a key task to execute tests in a disciplined way. Figure 8 shows this specific goal.

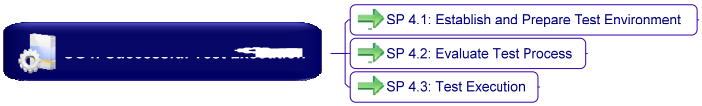


Figure 5: Specific Practices of SG 4.

Specific goal five has associated three specific practices. Testing process does not finish with test case execution, otherwise it would be necessary to evaluate testing process and define corrective actions. Figure 9 shows this specific goal.

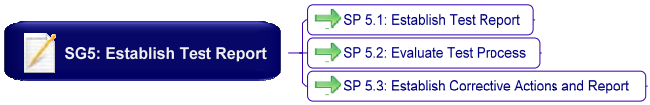


Figure 6: Specific Practices of SG 5.

Specific goals and practices have been described briefly in this section. A complete version of this process area is available at (http://sel.inf.uc3m.es/asanz/testpai/testpai_pa_fullversion.pdf). In this version, purpose, introductory notes, related process area and generic goals and their associated generic practices were defined as well.

4 Implementation of TestPAI

We implemented TestPAI in a real Spanish organization to verify that TestPAI improves the testing process and its integration with CMMI is easy and complete.

The company is one of the main insurance corporations of Spain which market is dealing with farmers, livestock men and insurance enterprises. It has about 200 employees and development department personnel 30 more or less (Sept. 2007).

Its mission is to provide solutions needed by the rest of organization areas in the company with a good quality level, and optimizing the delivery term for each assignment. The software developed by them is ERP's deployment and customization.

In a previous software process improvement initiative, the organization achieved CMMI level 2. The current business goal is to achieve CMMI level 3 but they are also interested in improving their testing process at the same time. They had increased their profits since they achieved CMMI level 2. So, they did not want to adopt another different philosophy. TestPAI solved their problem by means of its integration with CMMI.

We defined several steps to implement TestPAI in the organization. First, the improving goals were established. Next, the current practice of the organization was studied to determine good and bad practices. After that, we adapted the development processes to include the TestPAI practices. Finally, we evaluated the use and improvement of the testing process previously defined. Figure 7 shows these steps and the main activities of each one.

Moreover, we evaluated the TestPAI implementation by means of a verification process which is based on creating and using checklists. Verification process defines two verification types:

A. To verify that verification items have been carried out: In it, the SEPG checks that notified items have been really carried out. This verification can be done in two ways: automatically or by hand.

B. Verification of quality of carrying out verification items: Verifiying that items had been carried out in a right way.

Figure 8 summarizes the verification process and Table 3 shows an example of a checklist which has been used in the verification of TestPAI implementation.

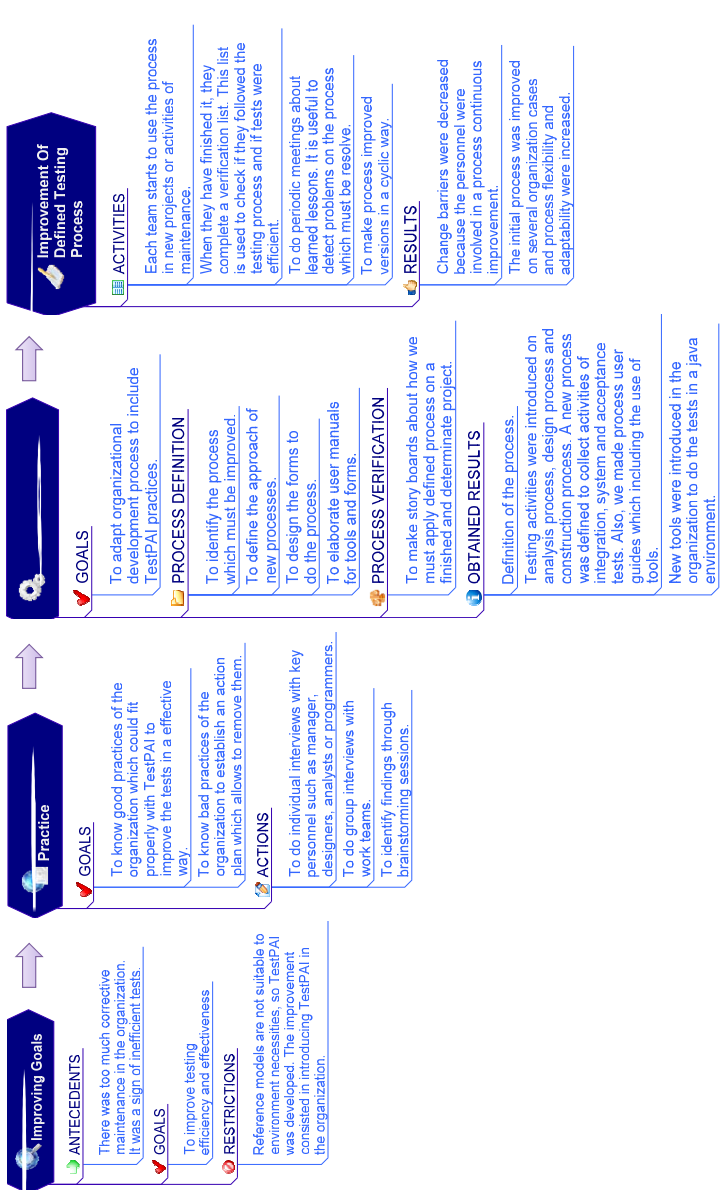


Figure 7: Implementation of TestPAI.

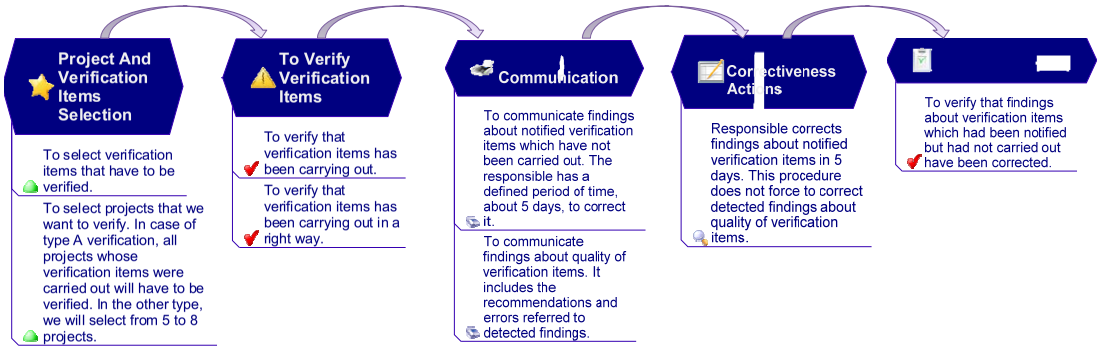


Figure 8: Verification Process.

Responsible:					
Identifier:			Work:		
Develop Environment:			Date :		Procedure
ITEM TO CONTROL	**ACCEPTANCE**	**OBSERVATIONS**	**CORRECTIVENESS ACTIONS**	**CLOSING DATE**	**LESSONS LEARNED**
If system sends or receives information from systems third party, the rules of record and design have been established.	☐				
Approval/rejection of established rules of record and design has been requested. The corresponding answer has been recorded.	☐				
The rules of record and design have been recorded on their corresponding folders and their publication has been requested.	☐				
Hardware and software environment have been specified to execute acceptance tests.	☐				

Table 3: A fragment of checklist for complementary activities to make the system design.

5 Conclusions

TestPAI is a detailed defined process area which is totally integrated with CMMI. This process area permits the improvement of testing process and other processes at the same time.

Many organizations use CMMI to improve their processes in Spain. In many cases, the testing process is important to their business, so they are interested on improving it. CMMI does not fully support the testing process. Therefore, it is not a suitable reference model for test improvement. TestPAI was developed to solve this problem. TestPAI was defined in compliance with CMMI, both staged and continuous representations.

TestPAI was implemented in a real organization. In order to do so, we defined a new testing process and adapted different development processes such as design and construction.

We implemented TestPAI in several pilot projects and we have concluded that previous training is necessary in testing activities and in tools which support these activities. We need to improve individual skills and competences of the staff involved in test activities.

6 Literature

[1] Kit, E. Software Testing in the Real World. Addison-Wesley. 1995

[2] CMMI Maturity Profile September 2007 Report. Software Engineering Institute. Carnegie Mellon University, September 2007 (http://www.sei.cmu.edu/appraisal-program/profile/pdf/CMMI/2007sepCMMI.pdf)

[3] Bush, M., Dunaway, D. CMMI Assessments. Motivating Positive Change. Addison-Wesley. 2005.

[4] García, J., de Amescua, A. Velasco, M., Sanz, A. Ten Factors that Impede Improvement of Verification and Validation Processes in Software Intensive Organizations. EuroSPI 2007. Postdam, Alemania. September, 26 - 28, 2007.

[5] CMMi® for Development. August, 2006.

[6] Paulk, M., Weber, C., Curtis, B., Chriss, M. The Capability Maturity Model. Addison-Wesley. Reading MA, 1995.

[7] Burnstein, I. Practical Software Testing. Springer-Verlag. 2002.

[8] van Veenendaal, E., Guidelines for Testing Maturity. Published in: STEN Journal, Vol IV, 2006.

[9] van Veenendaal, E., Test Maturity Model Integration (TMMi) Versión 1.0. (www.tmmifoundation.org). 2008.

[10] Koomen, T. Test process improvement : a practical step-by-step guide to structured testing. 1999.

[11] Pol, M., Teunissen,R., van Veenendaal, E.. Software Testing. A guide to the TMap Approach. Addison-Wesley. 2002.

[12] Koomen, T., van der Aalst, L., Broekman, B., Vroon, M. TMap Next for result-driven testing. UTN Publishers. 2006

Acknowledgments

This work is supported by the Spanish Ministry of Education and Science through the following projects: TIN-2004-07083 (GPS: Plataforma de Gestión de Procesos Software; Software Process Management Platform) and TIN2007-30391-E/ (REPRIS - Red para la promocion y mejora de las pruebas en ingenieria del software). It is also supported by PROGRESION SMP through the project named “Diseño de un Modelo de Referencia para la Gestión Estratégica Ligada a la Mejora de Procesos en Organizaciones Intensivas en Software” (UC3M 2006/03617/001).

7 Author CVs

Ana Sanz Esteban

She received an Engineering degree in Computer Science at Carlos III University of Madrid, and Msc in Computer Science and Technology by the same university. She is a PhD student in the Computer Science Department at the Carlos III University of Madrid. Her research on software engineering is focused on software process improvement, especially test process improvement. It involves different maturity models related with organization in general like CMMI as well as related only with test process like TMM or TPI. She has several years of experience as a software engineer.

Javier Saldaña Ramos

He received an Engineering degree in Computer Science at Carlos III University of Madrid, and Msc in Computer Science and Technology by the same university. He is a PhD student in the Computer Science Department at the Carlos III University of Madrid. His research on software engineering is focused on software process improvement through teamwork process improvement. It involves team management improvement, virtual teams management improvement and the use of TSP (Team Software Process) in teamwork. He has several years of experience as a software engineer.

Javier García Guzmán

He received an Engineering degree and PhD in Computer Science at Carlos III University of Madrid. He is software process improvement consultant in PROGRESION SMP. He has 9 years of experience as software engineer and consultant in public and private companies. He has participated in numerous research projects, financed with public (European and national) and private funds, in relation to software process improvement and its integration with the organizational business processes. He has published books and international scientific papers related to software engineering and collaborative working environments. His current research interest is formal measurement of processes improvement, ISO 15504 assessments, software capacity evaluations and audits and knowledge management related to software engineering.

Domingo Gaitero Gordillo

He received an Engineering degree in Computer Science at Polytechnical University of Madrid. He is responsible of IT Governance in ATOS ORIGIN Spain. He has several years of experience working in Software Engineering and Quality and Software Process Improvement. He has published a book related to the Spanish standard software development methodology for Public Administrations called METRICA. He participates in different symposiums and international and national conferences. Also, he takes part in committees and professional associations of IT.

Analysing the Cost of Lightweight SPI Assessments

Fergal McCaffery, and Gerry Coleman
Dundalk Institute of Technology,
Dundalk,
Ireland.
Fergal.mccaffery@dkit.ie, gerry.coleman@dkit.ie

Keywords:

Assessment findings, Software Process Improvement (SPI)

Abstract

In this paper we describe the implementation of an assessment method that was developed to assess software processes within small to medium-sized Irish software organisations that have little or no experience of software process improvement (SPI) programmes. We discuss the actual overheads associated with performing software process assessments based upon our experiences of performing assessments in three small to medium sized (SMEs) software development companies.

1 Introduction

For many small software companies, implementing controls and structures to properly manage their software development activity is a major challenge. A software process improvement (SPI) programme, properly managed and administered, can assist software companies in this regard. An SPI programme is best approached through the medium of a process assessment. Assessments help to highlight strengths and weaknesses in an organisation's processes and, thereby act as a catalyst for the SPI initiative.

However, assessments are an expensive business and typically require significant company resources to achieve meaningful impact. As a result, and to enable small companies gain the benefits of SPI, in 2006 we created the Adept assessment method [1]. Based on Capability Maturity Model Integration (CMMI) Class 'C' appraisal guidelines [2] and incorporating practices from ISO/IEC 15504 [3], Adept offers a lightweight mechanism for commencing, or kick-starting, an SPI programme. At present, 3 assessments have been carried out using the Adept method. Whilst the results and feedback on the method itself support the authors' beliefs that it has benefit for small companies, this paper reports on actual costs associated with performing this assessment within the three software SMEs.

This paper is structured as follows. Section 2 describes the software industry in Ireland including the types of companies that were assessed and provides an overview of the Adept method. Section 3 details the implementation of the Adept assessment within the three companies. Section 4 provides an analysis of the overheads associated with performing the assessments. Section 5 discusses the significance of these overhead costs and section 6 offers conclusions and a description of future work.

2 Adept Assessments and the Irish Software Industry

2.1 The Irish Software Industry

The software industry in the Republic of Ireland (ROI) is a key component of the national economy. According to Enterprise Ireland (EI) (ROI's economic development agency for indigenous companies) at the end of 2004, Irish-owned software businesses comprised over

750 companies employing almost 12,000 people [4]. The majority of these Irish-owned software firms are small where it is calculated that only 1.9% employ more than 100 people whilst more than 60% employ 10 or fewer [5]. The venture capital group, HotOrigin produce an annual report on the state of the indigenous software industry and estimate there is a total of 417 indigenous software product companies in Ireland [6]. They categorise these companies across three stages of development, 'Start-up' (1-25 employees), 'Build' (26-75 employees), and 'Expansion' (75+ employees). The 2004 report shows that almost three-quarters of indigenous software firms fall into the Start-up category, with about 9% in the Expansion category and the remainder in the Build category.

The profile in Northern Ireland (NI) is similar. A Mintel study [7] on the software sector in NI showed that, at the beginning of 2005, there were 348 companies employing approximately 5,500 people. Of these 348, 71% are indigenous but account for only 12% of the staff. However, over 75% of the companies employ fewer than 15 people.

Because of the industry profile, and the prevalence of small and very small organisations, the need for an assessment method that requires very little client resources is clear. To support industry improvement, and to promote competitiveness and success of the indigenous software industry of the entire island of Ireland, both EI and Momentum (the Northern Ireland ICT federation) are currently engaging member companies in SPI initiatives.

This paper details how Adept assessments were performed in three indigenous software development organisations with the ROI. These companies employed approximately 5, 22 and 100 software developers and this therefore provides a good sample of Irish software SMEs. None of the companies had any prior experience of software process improvement or particularly of either SW-CMM or CMMI. The three companies also work in three different sectors namely, financial, medical and space technologies.

2.2 Overview of the Adept method

Adept was developed specifically to encourage Irish SMEs to improve their software development practices. Enterprise Ireland (representing the Irish SMEs), requested that Adept take the following factors into account:

- Improvement is more important than certification and a rating is not required;
- The amount of preparation time required by the company for the assessment should be minimal;
- The assessment should be performed over a short period of time;
- The assessment method should enable companies to select assessment in process areas that are most relevant to their business goals;
- Whilst the assessment will be based upon both the CMMI® [8] and ISO/IEC 15504 [9] models the SPI models should be invisible to the SMEs that are being assessed.

As the main aim of the Adept method is to encourage SPI improvement based upon the generic SPI principles that are shared by both CMMI® and ISO/IEC:15504 we decided to base the Adept method upon relevant process areas from the CMMI® model and include input from the ISO/IEC:15504 model. Therefore the Adept method consists of an assessment component for each CMMI® process area that is deemed applicable for Irish SMEs. However, even though each assessment component adopts a CMMI® process area name, it will provide equal coverage of both the CMMI® and ISO/IEC 15504 models by containing questions that relate to ISO/IEC 15504 in addition to questions based upon the CMMI® model.

The process areas included in Adept are based upon the results of previous research performed by DkIT (Dundalk Institute of Technology) and Lero in relation to software processes within Irish SMEs [10, 11, 12]. Based upon this information the following 12 process areas were selected: Requirements Management, Configuration Management, Project Planning, Project Monitoring & Control, Measurement & Analysis and Process & Product Quality Assurance, Risk Management, Technical Solution, Verification, Validation,

Requirements Development and Product Integration.

Any of these 12 process areas may be assessed using the Adept method. However, four are classified as mandatory - Requirements Management, Configuration Management, Project Planning, Project Monitoring & Control. The choice of mandatory process areas was based upon the overlap of three factors. Firstly, priority was given to the CMMI level 2 process areas as these are deemed to be the foundation upon which a successful software development company should be based. Secondly, priority was assigned to the process areas in which SMEs would gain most benefit [13]. Thirdly, research in Ireland has shown that specific processes are seen as important by software SME managers [10, 11, 12].

3 Performing the Adept assessments

Currently, we have performed an Adept assessment in three software SMEs. As cost and time issues were important for each of the companies, we restricted the on-site interviewing to one day in all cases. This meant that the assessment was limited to a maximum of six process areas as this is as many as can reasonably be covered within one day [13]. Therefore, in addition to being assessed in the four mandatory process areas, companies will also be able to choose two of the other process areas. However, one of the three companies only desired an assessment in the four mandatory process areas. The process areas assessed within the three companies are shown in Table 1.

Table 1. Process areas assessed

Company A	**Company B**	**Company C**
Requirements Management	Requirements Management	Requirements Management
Project Planning	Project Planning	Project Planning
Project Monitoring & Control	Project Monitoring & Control	Project Monitoring & Control
Configuration Management	Configuration Management	Configuration Management
Technical Solution	Verification	
Product Integration	Risk Management	

The complete Adept method consists of eight stages, however stages 7 & 8 have yet to be performed in any of the assessed companies. In each of the assessments the assessment team consisted of two assessors who collaboratively performed the assessment. The stages were performed as described below:

Stage 1 (*Develop Assessment Schedule and Receive Site Briefing*). During this stage the assessment team met with senior management from the software company wishing to undergo an SPI assessment. The assessment team had two objectives during this stage. The first objective was to agree upon an overall schedule for the assessment and to select the six most applicable process areas. The time taken to achieve this objective varied from 30 minutes to 80 minutes across the 3 companies (see Table 2), ranging from a senior management that had prior knowledge of SPI and were committed to improving process areas that the assessment team deemed suitable, to senior management that required convincing of the benefits of SPI prior to agreeing upon an assessment or those disagreeing with the assessment team in terms of the process areas that should be assessed.

The second objective of the assessment team was to gain an understanding of the organisation to be assessed. Two of the companies provided this information solely by answering questions that were posed by the assessment team. The other organisation provided this information using a presentation combined with a question and answer session at the end of the presentation. Whilst both formats achieved the desired objective the assessment team liked the presentation approach as it provided them with structured

information in relation to the organisation and this enabled them to seek additional clarification. However, as time is a key factor in relation to performing an Adept assessment it is important that the presentation is focused in the areas of the company's structures, history, business objectives, types of ongoing projects, along with the lifecycle stage that each project has reached. If the presentation is not focused in these areas then such a presentation would be a waste of time both for the assessment team and the company (both in terms of preparation and presentation time), particularly as time would have to be spend afterwards in a lengthy question and answer session. This session lasted between 30 to 70 minutes (see Table 2) and this time really depended upon the size of the organisation, with smaller companies having fewer projects and organisational structures (e.g. Company C) to describe.

Table 2. Overheads associated with performing stage 1 of Adept

Company	A	B	C
Objective 1 : Assessment schedule meeting duration (mins)	30	80	50
Objective 2 : Site Briefing duration (mins)	50	70	30
Time taken for stage 1 meetings (mins)	80	150	80
No. of Assessors	2	2	2
No. of Company Participants	1	1	3
Total Assessor time for stage 1 (mins)	160	300	160
Total Company time for stage 1 meetings	80	150	240

This stage involved 2 assessors for each of the three companies. One senior manager participated in this stage for 2 of the companies while one senior manager and two project managers participated for the other company. Therefore the (stage 1) assessor time for each company was between 160 and 300 minutes, and the company employee time was between 80 and 240 minutes. In case of company C, they viewed the assessment as an opportunity for staff training and were keen that as many staff as possible participated.

During stage 2 (*Conduct Overview Briefing*) the lead assessor provided an overview of the Adept method to everyone within the appraised organisation who participated in various stages of the assessment. This session was key to the success of the assessment as it removed any concerns that the company personnel had in relation to both their role within the assessment and the confidentiality of their input. This overview session involved 2 assessors and between 5 to 10 company staff from the assessed organisations. In all cases it lasted approximately 1 hour (see Table 3). The same presentation was given to all three companies but the times varied slightly due to some companies asking more questions than others.

Table 3. Overheads associated with performing stage 2 of Adept

Company	A	B	C
Stage 2 : Conduct Overview Briefing (mins)	45	70	55
No. of Assessors	2	2	2
No. of Company Participants	7	10	5
Total Assessor time for stage 2 (mins)	90	140	110
Total Company time for stage 2 meetings	315	700	275

Stage 3 (*Analyse Software Documentation*) provided the assessors with a brief insight into project documentation. As Requirements Management, Project Planning, Project Monitoring and Control, and Configuration Management were assessed in each of the 3 organisations

the assessors requested the same type of documentation from each company. This consisted of: a typical project plan, a typical project progress report, a typical approved requirements statement and any documentation relating to the company policy on configuration management. During this stage the assessors scanned documentation firstly for existence, and secondly for structure as opposed to content. This stage was really used to develop questions based upon the documentation for stage 4.

This stage involved 2 assessors and 1 member of personnel from the assessed organisation who provided access to the documentation. Typically, the company member dedicated 1 hour to retrieving the requested documents. When the authors originally created the Adept method they proposed that the 2 assessors would each analyse the documentation for approximately 3 hours. Upon performing the actual assessment the authors discovered that 1.5 hours was adequate. Therefore we discovered that 3 person-hours of assessor time and 1 person-hour of company time are sufficient for this stage.

Stage 4 (*Conduct Process Area Interviews*) is the main part of the Adept assessment. In this stage nominated staff members (those deemed by management within the assessed organisation to be knowledgeable in relation to a particular process area) were interviewed.

In two of the companies 6 interviews were performed, with four being performed in the third company.

Table 4. Overheads associated with performing stage 4 of Adept

Company	A	B	C
Process area interview time (mins)			
Requirements Management interview time (mins)	65	70	80
No. of Company Participants	1	3	4
Company time for Requirements Management interview	*65*	*210*	*320*
Project Planning interview time (mins)	80	90	105
No. of Company Participants	3	3	4
Company time for Project Planning interview	*240*	*270*	*420*
Project Monitoring & Control interview time (mins)	75	85	100
No. of Company Participants	3	3	4
Company time for Project Monitoring & Control interview	*225*	*255*	*400*
Configuration Management interview time (mins)	45	55	70
No. of Company Participants	2	3	4
Company time for Configuration Management interview	*90*	*165*	*280*
Technical Solution interview time (mins)	55	N/a	N/a
No. of Company Participants	2	N/a	N/a
Company time for Technical Solution interview	*110*	*N/a*	*N/a*
Product Integration interview time (mins)	50	N/a	N/a
No. of Company Participants	2	N/a	N/a
Company time for Product Integration interview	*100*	*N/a*	*N/a*

Verification interview time (mins)	N/a	65	N/a
No. of Company Participants	N/a	3	N/a
Company time for Verification interview	*N/a*	*195*	*N/a*
Risk Management interview time (mins)	N/a	55	N/a
No. of Company Participants	N/a	3	N/a
Company time for Risk Management interview	*N/a*	*165*	*N/a*
Total *Company interview time (mins)*	830	1260	1420
Interview time (mins)	370	420	355
No. of Assesors	2	2	2
Total Assessor time (mins)	740	840	710

When the authors developed the Adept method they predicted that each interview would last approximately 1 hour. However after performing three assessments it now appears that more time should be devoted to certain process areas than others. Table 4 illustrates that the project planning and the project monitoring & control process areas typically require more time than the other process areas and typically require 1.5 hours, therefore 7 hours should be scheduled in order to complete the 6 process area interviews. It should also be noted that Company C was assessed in 4 process areas as opposed to 6 process areas. Company C desired an assessment that would not only enable the chosen process areas to be assessed but they also wanted their staff to be able to ask questions of the assessors within each of the process area interviews. To enable this to be achieved within a one-day on-site timeframe it was mutually agreed, between the assessors and the senior management of company C, that only the four mandatory process areas would be assessed. Table 4 illustrates that the individual process area interview were therefore longer for company C than the other two companies. It can also be noted from Table 4 that company A dedicated as few company participants as possible in order to perform the assessment whereas company C viewed the assessment as an opportunity to educate staff and therefore 4 staff out of a total staff pool of 7 participated in each interview session. It can also be noted from Table 4 that in the case of the process area interviews that were common (the same set of scripted questions were used) to both Companies A and B that all the interviews lasted longer for company B. In 2 out of the 4 interviews company B had more participants and this could have impacted the duration of the interview, however whenever the interviews contained the same number of participants this was also the case. The authors report that this was due to the fact that Company B was a larger organisation and had more ongoing projects and therefore more project teams and a greater number of practices within each process area and therefore this required a more extensive question and answer session.

Two assessors participated in each process area interview. This equated to between 12 person-hours and 20 minutes of assessor time for a company A, 14 person-hours of assessor time for company B and 11 person-hours and 50 minutes of assessor time for company C. In relation to company time 13:50, 21 and 23:40 person-hours were consumed for companies A, B and C respectively.

When performing the assessments the authors also scheduled the process area interviews in the natural sequence of the software development lifecycle. This helped the assessors to develop an accurate profile of the software development practices adopted by a company and enabled the responses from process areas' interviewees earlier in the day to be cross-referenced in later interviews.

During each of the process area interviews, the lead assessor asked a combination of pre-defined and follow-up questions while the other assessor made notes. Additionally the lead

assessor used an Excel based tool to make an initial judgement about the responses by judging them against a discrete set of values – Red (not practiced), Amber (partially practiced), Yellow (largely practiced) and Green (fully practiced). This enabled the opinions of the questioner and not just the note-taker to be recorded for subsequent review in stage 5.

Stage 5 (*Generate Appraisal Results and Create the Findings Report*) was an off-site stage. This involved a both of the assessors working together to agree upon a findings report. The output of this stage was a findings report for each company that consisted of a list of strengths, issues and recommendations for each of the assessed process areas. The findings reports were developed through the assessors firstly focusing upon the scores produced by the Adept tool and initially searching for particular areas of extreme (i.e. All red or all green) and then reviewing the associated interview notes and the for each of the 6 assessed process areas. This stage involves 2 assessors collaborating together for 5:30, 6 and 4:30 hours respectively for Company A, B & C. As only 4 process areas were assessed in company C the production of the findings report took considerably less time. It seemed to take approximately 1 hour to document the findings report of each process area. Therefore a total 11,12 and 9 person-hours of assessor time was required for these stage in relation to companies A, B and C respectively.

Stage 6 (*Deliver the Findings Report*) involved presenting the findings report to the staff in the assessed organisation who participated in the interviews (see Table 5). Both assessors participated in these presentations. The briefing session lasted between 40 to 70 minutes for each of the three assessed companies. As there was only 4 process areas assessed in company C its findings presentation was the shortest. The findings presentation for company B was the longest due to the fact that more projects were assessed and more staff asked questions during this presentation than in any of the other company presentations.

Table 5. Overheads associated with performing stage 6 of Adept

Company	**A**	**B**	**C**
Stage 6 : Deliver the findings report (mins)	50	70	40
No. of Assessors	2	2	2
No. of Company Participants	7	10	5
Total Assessor time for stage 6 (mins)	100	140	80
Total Company time for stage 6 meetings	350	700	200

Stage 7 (*Develop a SPI Path with the Company*) involves collaborating with staff from the appraised company to develop a roadmap that will provide guidance to the appraised company in relation to practices that will provide the greatest benefit in terms of the company's business goals. This stage was not performed with any of the three companies.

Stage 8 (*Re-assess the SPI Path and Produce a Final Report*) involves revisiting the appraised company approximately 6 months after the completion of stage 7 and reviewing progress against the SPI path that was developed in stage 7. The outcome of this stage will be an updated SPI path and a final report detailing the progress that has been accomplished along with additional recommendations. This stage was not performed with any of the three companies.

4 Analysis of Assessment Effort

In comparing the results in Table 6, and to ensure a like-for-like comparison, we have used only the 4 process areas in Stage 4 that were common to the 3 assessed companies.

Table 6. Summary of the Adept Overheads for the Three Companies

Adept Assessment Stage	Process Area	Company Time (CT) /Assessor Time (AT) Spent (Mins) in Assessment Stage (by Company)					
		A		B		C	
		CT	AT	CT	AT	CT	AT
Stage 1 - Develop Assessment Schedule and Receive Site Briefing		80	160	150	300	240	160
Stage 2 - Conduct Overview Briefing		315	90	700	140	275	110
Stage 3 - Analyse S/W Documentation		60	180	60	180	60	180
Stage 4 - Conduct Process Area Interviews	*Requirements Management*	65	130	210	140	320	160
	Project Planning	240	160	270	180	420	210
	Project Monitoring & Control	225	150	255	170	400	200
	Configuration Management	90	90	165	110	280	140
Stage 5 - Generate Appraisal Results and Create Findings Report			660		720		540
Stage 6 - Deliver Findings Report		350	120	700	120	200	120
Total Time (Mins)		**1425**	**1740**	**2510**	**2060**	**2195**	**1820**

All other Adept stages, 1–3 and 5–6, were common to all 3 companies. In carrying out the assessments we can see that the amount of company time involved ranged from 1425 minutes (c.24 hours) to 2510 minutes (c.42 hours) averaging at 2043 minutes (c.34 hours). Company B generated the highest value as the greatest number of its participants took part in the process. If we assume a company utilisation rate of 7 hours per staff day, then the average commitment for the 4 process area assessments, c.34 hours, is equivalent to 1 staff week.

The range of assessor time is narrower, 1740–2060 minutes (29-34 hours), as the differences can be explained by company A having the fewest number of employee participants, which reduced the number of queries directed to the assessors, and company B having the largest number of participants which generated the greatest number of questions. Assuming a similar utilisation rate for an assessor, 7 hours per staff day, then the assessor cost average for the 4 process area assessment, 31 hours, is also equivalent to one staff week.

Based on these figures we can attempt to determine the actual cost for the 4 process area assessment. Determining an accurate cost for the assessment clearly depends on the company's own costing model, and the assessor fee. If we take a company cost figure of €350 euros per staff day, and an assessor cost of €750 euros per day, we arrive at a total cost for the assessment of €5500. This figure is however, something of an underestimate as it includes only time costs and makes no provision for travel and hotel expenses for the assessors which will likely be incurred during the assessment. It also does not include the cost for Adept stages 7 and 8 which involve working with the company on developing an SPI initiative based on the assessment results and the subsequent follow-up visit to review the success, or otherwise, of that SPI activity.

5 Discussion on the Significance of the Overhead Costs

There is little information contained in the literature regarding the actual cost of engaging in assessment-based SPI, especially at the small company level. Most of what is reported relates to larger scale CMM or CMMI SCAMPI-based appraisal. However, the cost of implementing CMM/CMMI-based improvement can be very high, from in excess of 100,000 dollars [14], 180 person-days on process redefinition, 70 person-days on training and 20 person-days on evaluations on a programme to secure CMM Level 2 accreditation [15]), to 45,000 dollars for the initial assessment activities and 400,000 dollars to move from level 2 to level 3 [16]. Herbsleb *et al.* [17] conducted a multiple case study of companies who had experienced success with CMM. However, the majority of those companies felt that implementing CMM had cost more than expected, leading the SEI-employed authors to concede that CMM is neither a cheap nor quick fix. In addition, CMM can negatively impact a small software company's competitive potential [18]. Brodman and Johnson, [19] report a number of resource-related difficulties that small companies have in attempting to implement the CMM, a fact conceded by the CMM's own proponents [20].

The heavy expenditure required to engage in full CMM/CMMI-based improvement, documented above, demonstrates the difficulties that small companies face in trying to engage in SPI. Most small companies have very limited finances and so a resource-intensive SPI initiative is clearly unattractive. Our objective in developing Adept was to provide a cheaper SPI alternative to smaller organisations. The feedback in this regard from the companies assessed has been largely positive both from the economic perspective regarding assessment cost and the actual benefits which companies felt flowed from the process and subsequently.

6 Conclusion and Future Work

This paper aims to show the true nature of SPI assessment in small software companies and the costs incurred by the companies involved. All of the companies reported major benefit from the assessments which have been carried out within the last year. It is our intention to make further contact with the companies concerned in the near future to see how their SPI activity has progressed since the assessments. We believe that by showing the real cost of engaging in assessment-based SPI, companies can be more informed about undertaking initiatives and can plan and budget accordingly. Only by doing a true cost-benefit analysis in this way can companies measure the success of SPI initiatives.

We plan to continue to promote the Adept approach and intend to expand the offering through e-Adept, a web-based service, which will enable company self-assessment, with remote assessor support, using the Adept framework. Initial company discussions indicate significant interest in an on-line assessment programme which can reduce assessment formality and cost even further, and allow more regular company, team or project self-examination.

Acknowledgement

This research has been funded by Science Foundation Ireland through the Stokes Lectureship Programme, grant number 07/SK/I1299.

References

[1] McCaffery, F., Taylor, P. And Coleman, G., 'Adept: 'A Unified Assessment Method for Small Software Companies', *IEEE Software*, Jan/Feb 2007, pp. 24-31.

[2] Software Engineering Institute: CMMI Product Team, 'Appraisal Requirements for CMMI, Version 1.1 (ARC, V1.1)', *Technical Report CMU/SEI-2001-TR-034, ESC-TR-2001-034*, 2001; http://www.sei.cmu.edu/pub/documents/01.reports/pdf/01tr034.pdf

[3] ISO/IEC 15504: 'Information Technology – Process Assessment – Part 5: An exemplar Process Assessment Model', ISO/IEC JTC1/SC7, October 2003.

[4] Enterprise Ireland, 2005b, 'Software Industry Statistics 1991-2004', available at http://www.nsd.ie/htm/ssii/stat.htm.

[5] Crone, M., 2002, 'A Profile of the Irish Software Industry', Northern Ireland Economic Research Centre (NIERC), Belfast NI.

[6] HotOrigin, 2004, 'Ireland's Software Cluster: Preparing for Consolidation', HotOrigin Ltd., Dublin, Ireland.

[7] Available through Mintel - www.mintel.com

[8] 'Capability Maturity Model® Integration for Development', Version 1.2 (2006), http://www.sei.cmu.edu/publications/documents/06.reports/06tr008.html, Technical Report CMU/SEI-2006-TR-008

[9] 'ISO/IEC 15504-3:2004 - Information Technology - Process assessment - Part 3: Guidance on performing an assessment' – **Need full ref here!**

[10] Coleman, G., 'An Empirical Study of Software Process in Practice', in *Proceedings of the 38th Annual Hawaiian International Conference on System Sciences*, - Big Island, HI, p. 315c. January 2005.

[11] Meehan, B. and Richardson, I., 'Identification of Software Process Knowledge Management', *Software Process: Improvement and Practice*, Volume 7, Issue 2, June 2002, pp 47-56.

[12] Blowers, R. and Richardson, I., 'The Capability Maturity Model (SW and Integrated) Tailored in Small Indigenous Software Industries', International Research Workshop for Process Improvement in Small Settings, Software Engineering Institute, Pittsburgh, Pennsylvania, U.S.A., October 19-20, 2005. http://www.sei.cmu.edu/publications/documents/06.reports/06sr001.html.

[13] F.G. Wilkie, D. McFall & F. Mc Caffery, 'An Evaluation of CMMI Process Areas for Small to Medium Sized Software Development Organisations', *Software Process Improvement and Practice*: Volume 10, Issue 2, June 2005, pp. 189-201 - Wiley Publishers.

[14] Saiedian, H., and Carr, N., 'Characterising a Software Process Maturity Model for Small Organisations', in *ACM SIGICE Bulletin*, Vol. 23, No. 1, July 1997, pp. 2-11.

[15] Kelly, D.P. and Culleton, B, 'Process Improvement for Small Organisations', in *IEEE Computer*, October 1999, pp. 41-47.

[16] Humphrey, W.S., Snyder, T. and Willis, R., 'Software Process Improvement at Hughes Aircraft', in *IEEE Software*, July 1991, pp. 11-23.

[17] Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W. and Paulk M., 'Software Quality and the Capability Maturity Model', in *Communications of the ACM*, Vol. 40, No. 6, 1997, pp. 30-40.

[18] Bach, J., 'The Immaturity of CMM', in *American Programmer*, September 1994, Vol. 7, No. 9, pp. 13-18.

[19] Brodman, J.G. and Johnson D.L., 'What Small Businesses and Small Organisations say about the CMM', in *Proceedings of the 16th International Conference on Software Engineering*, 1994, pp. 331-340.

[20] Paulk, M., 'Using the Software CMM in Small Organisations', in *Proceedings of 16th Pacific Northwest Software Quality Conference*, 1998, pp. 350-361.

Crossing the Border between Process and Product -The Requirements Catalogue Maturity Model

Tomas Schweigert
Stollwerckstr. 11
51149 Köln
Tomas.schweigert@SQS.de

Abstract

At the current state of research and discussion product and process characteristics seem to be unconnected. This is reflected by the structure of standardisation organisations, conferences and sources. This doesn't reflect the interlocking between product and process. The product inherits characteristics of the process. This leads to the question if the measurement framework of process quality is transferable into a suitable measurement of some aspects of product quality. The paper includes a first idea of this transfer. The description of this idea uses the requirements as an example which shows the difficulties and the benefits of this approach. As a result the maturity model for a requirements catalogue is outlined and the benefits for the management are explained.

Keywords

Requirements, Maturity models, Process Capability, Measurement

1 Introduction

There are several models in place to describe the quality, content an process of requirements (ISO 9126, ISO 15504, IEEE 830, IEEE 1233). These models address different aspects of requirements management and requirements engineering. There are also some researchers who set up an own measurement frame for the requirements engineering process [Hood 05] which contains product characteristics derived from IEEE 830.

This leads to the question why we need to come to a better understanding of requirements and the requirements process.

As we look to the Chaos Report we see that inadequate requirements create a high risk of project failure. But can an inadequate requirement lead to project failure? Not direct. But it might lead to wrong decisions of the management.

In fact we do know: If requirements are fluffy or unknown the project is research and should adopt an agile approach if the requirements are clear the project is industry and should use a waterfall or a V-model. So requirements can't be inadequate by them self but might be inadequate for the chosen approach, the budget or the delivery date.

Does the current development of requirements management solve this issue? No because insufficient requirements are insufficient requirements even if they are written in UML.

I have seen and analysed some failed projects. Some of these projects had an immature requirements catalogue. Most with inconsistent detailing level

....

- The System must be able to administrate our customers

- Customer name has to be written in light bold italic letters.

...

Such catalogues seem to be very precise. But they are nothing more than an initial collection of ideas resulting of a brainstorming. It is clear, that these type of requirement catalogue creates a high risk that the management comes to wrong decisions about approach, budget or plans.

Wouldn't it be nice -from a management perspective- to have a characteristic which shows the maturity of a requirements catalogue and supports decision making and risk analysis?

2 Requirements as a 1st Example of the exchange between Process and Product Characteristics

When we talk about the exchange of characteristics between process and product, and look of a potential first example, we should look after a product with defined characteristics from the process and the product view at the one hand and a high range of criticality at the other. Requirements are also important for management and engineering purposes.

Requirements are known as a critical success factor. Inadequate requirements are one of the most often assigned reasons for project failure [Standish 95-06 Gilb04 Centreline04].

To develop a requirements catalogue is time and budget consuming. Normally the IT management knows the cost of developing a requirements catalogue, but has only believes about the benefits of doing so.

What is missing is a characteristic that helps the management to understand the current situation and the resulting options.

Regarding this, the requirements might be a good example to test if a characteristic from a process model, the maturity, might also be a characteristic of the product.

3 Characteristics of the classical View on requirements

When we want to evaluate if we can define a maturity level for requirements catalogues we should have a look at the characteristic models that are in use for requirements.

3.1 Product: ISO 9126

ISO 9126 is a standard that defines the potential characteristics of a software product like correctness, performance maintainability and so on. Doing so the standard also defines the expected content of an requirements catalogue.

Addressing the structure of the content of an requirements catalogue ISO 9126 gives a formal criteria for completeness but no measurable characteristic or indicator for the quality or maturity of an requirements catalogue.

3.2 Process

3.2.1 ISO 15504

ISO 15505 addresses the capability of processes. There are two aspects in doing so: The process reference model and the measurement framework.

In the process reference model (Part 5) there are the following processes included:

- Requirements elicitation
- System requirements analysis
- Software requirements analysis

The measurement framework defines a rating scale with 6 levels:

- 0: Incomplete
- 1: Performed
- 2: Managed
- 3: Established
- 4: Predictable
- 5: Optimising

Based on 9 Process Attributes

- PA 1.1 Process performance attribute
- PA 2.1 Performance management attribute
- PA 2.2 Work product management attribute
- PA 3.1 Process definition attribute

- PA 3.2 Process deployment attribute
- PA 4.1 Process measurement attribute
- PA 4.2 Process control attribute
- PA 5.1 Process change attribute
- PA 5.2 Continuous improvement attribute

Each of these attributes addresses the capability of the process to increase performance and to avoid risk. In fact, this model is very generic. It neither does address the content of requirements nor does it address the quality of requirements.

3.2.2 The HOOD Capability Model

The Hood RME capability model also addresses the requirements process [Hood05]

Hood defines 6 Areas

- Scope
- Modelling
- Elicitation
- Specification
- Analysis
- Review

Which can reach 3 capability Levels (1..3)

The Model defines 4 Steps to reach Level 1 and 3 Steps to reach Level 3

Even if the model is focused on process it contains characteristics of product quality. The model states that requirements must be

- Atom
- Identifiable
- Provable
- Understandable
- Feasible
- Unambiguous
- Consistent
- Complete
- Free of redundancies
- Correctly developed
- Traceable to source

Looking at this we find the model somewhere between pure product and pure process.

3.3 Requirement

IEEE 830

IEEE 830 defines how requirements should be elicited and documented.

IEEE states the following quality criteria for requirements:

- Correctness: A requirement specification should describe what the software should meet according to the users' needs.
- Unambiguousness: A requirements specification should not allow multiple interpretations.
- Completeness: A requirements specification should include all significant requirements and all the outputs for all the inputs.
- Consistency: There should be no conflicts between requirements.
- Ranking of importance and stability: Requirements should be described with the indication of importance and stability.
- Verifiability: There should be finite feasible steps of process to check if a requirements specification satisfies all the desired quality properties.
- Modifiability: A requirements specification should be well structured and should not be redundant so that modified requirements do not result in any inconsistency.
- Traceability: Requirements should be traceable from or to documents of other requirements engineering activities.

1. Understandability: Requirements specification should help easy communication between stakeholders.

The standard does not define a measurement for requirements but there was a measurement defined by Chris Rupp et al. [Rupp06]. This measurements create no own model but allow to evaluate and quantify the quality of a requirements catalogue.

As a result the method delivers a spider web diagram which is a little helpful to understand some aspects of the quality of a requirements catalogue.

3.4 Other Models

The study "A Maturity Model of Evaluating Requirements Specification Techniques" seems from the title to handle the issues of this paper. The distinguishing is that this paper is focussed on management needs while the study from Yonghee Shin is on the one hand focussed on technology using the model from the Redwine/Riddle's Software Technology Maturity Model which contains 6 levels to technology maturity:

- 1. Basic Research
- 2. Concept Formulation
- 3. Development and Extension
- 4. Internal Enhancement and Exploration
- 5. External Enhancement and Exploration
- 6. Popularization

On the other hand the study is a little focused on requirements using the Pohls requirements engineering framework with the dimensions

- Specification
- Representation
- Agreement.

Her own model has fife levels of the goal-based technology maturity are defined as follows:

1. Need: A problem is identified and people recognize the necessity of a technology. Basic ideas are invented and published.

2. Attempt: Most of the partial solutions are developed and they are used only in the development group.

3. Usable: Most of the partial solutions are developed and they are used outside the development group.

4. Useful: Most of the partial solutions have commercial product quality and they are broadly used.

5. Indispensable: All of the partial solutions have commercial product quality and they are broadly used. This means that the goal of the technology was achieved.

The result of this model is an overall view on technology issues. Shin comes to the result that in her model the technology maturity of the requirements engineering is somewhat like 3.

The model does not deliver the maturity of a concrete requirements catalogue.

5 The Management View on Requirements

From the management perspective we find some key questions:

1) What is the decision which avoids risk and increases the probability of project success?

2) What are the probable consequences of a decision for budget, duration, resources or quality?

3) What are the necessary steps to implement a decision?

4) How to deliver the needed value with high efficiency?

For some of these questions requirements are a critical issue.

- Requirements are needed to estimate the attributes of the project.
- One important attribute is the budget needed to complete the work of the project
- Another attribute is the delivery date.
- Requirements might include scope creeping or gold plating
- There might be unpredictable changes in the requirements.
- When we look at the available sources of requirements engineering or requirements management we find, that some issues are well addressed:
- The formulation of requirements according to IEEE 830
- The stakeholder involvement
- The control of changes

If this is enough: why are requirements still a main source of project failure? Why run projects in trouble even if they have well formulated and atomic requirements?

Let's look at another source of project failure: Lack of management commitment [Standish 95-06] What is the link between requirements and management commitment? Could it be understanding? If yes: Are requirements understandable from a management perspective? The answer is: No. A big project has probably thousands of atomic requirements understandable only for subject matter experts. The first understandable data which the senior management receives are the estimates, the

budget and perhaps the business case.

There is no information about potential risk sources in the requirements catalogue and there is no support for making or review decisions.

There are metrics in the requirements engineering but they seem to be descriptive and not made to support decision making.

If requirements should be a basis for project success, there should be a measure which characterises the whole requirements catalogue, and supports the making of decisions. Which means if the requirements catalogue has a defined quality should the management decide to spent budget for improving the requirements or choose a development approach suitable for given quality of the requirements.

If we are searching for such a quality metric it should have some characteristics:

- Easy to understand
- Compatible with other standards
- Integrating other models.

What we can learn from the success story CMM/CMMi is that a one digit measure with clear defined levels fits best.

Conclusion: If a requirements maturity level can be defined it would help the management

- To understand the basis of given estimations and plans
- To understand the risk
- To understand the options and to review the given decisions.

4 The Requirements Catalogue Maturity Model

As shown there is a need to have the possibility to characterise a requirements catalogue by using a one digit scale. The scale and the context of each level should reflect known best practise and proven characteristics.

4.1 Inputs

The model is based on inputs from the process improvement area and from the requirements engineering discipline.

It combines both areas to deliver a clear picture over the quality of a requirements catalogue

4.1.1 Inputs from The ISO 15504 Measurement Framework

The ISO 15504 brings the idea of a measurement framework which helps to get a clear view on the capability of a process.

Why not adapt the idea to a product focus and define maturity levels for products like a requirements catalogue.

Why not use the techniques of describing capability levels in ISO 15504 to describe maturity levels for products like requirements.

4.1.2 Inputs from the requirements Engineering Discipline

It is clear that if we talk about the maturity of a product, we can't use the process reference model (PRM) of ISO 15504. What we need is content from the requirements engineering discipline like IEEE 830.

4.2 The Model

The model is based on 6 levels and 9 product attributes. The product attributes reflect typical requirements for single requirements or a requirements catalogue.

It also reflects the capability of key processes bringing the usage of best practices into the maturity of a requirements catalogue.

4.2.1 Levels

If we think about maturity levels for requirements we should think what degree of completeness un ambiguity or consistency might be reached by such a document.

But the descriptive part is only one aspect. The thesis is that each requirement maturity level may be combined with one ore more procedure models and indicates a typical level and content of risk.

Level 0: Bubble

At Level 0 the requirements are neither precise nor consistent in detailing level description technique they are just ideas or i.E the unrefined result of a brainstorming.

It is not clear who are the relevant stakeholders and their needs and the internal and external interfaces are not evaluated.

The users of the requirement might not understand what the requirement means or has a wide area of interpretation.

Level 1: Described

At level 1 Requirements are described in a way that the user of the requirements is able to develop an idea about possible solutions. The description is so clear an detailed that the estimation of the project planning and the further detailing and clarifying of requirements is possible.

At level 1 the relevant stakeholders and the external interfaces are identified and documented.

Communication with stakeholders is initial planned and running.

Stakeholders contribute to the documentation of the requirements.

Level 2: Precise

At level 2 the described requirements are precise. That means that their description is accurate so fulfilling the criteria of IEEE 830 and that they include acceptance criteria as a basis for test planning and test case development.

The precise formulation of the requirement supports verification and validation.

Bidirectional traceability is vertical and horizontal is supported.

Level 3: Standardised

At level 3 the precise requirements are standardised and measured. This means that the description of the requirements is based on an organization wide standard which covers the criteria of international standards like IEEE 830 or ISO 9126 and enhances the common understanding of the structure of the requirements catalogue and the description of each requirement. There are also data from quality measures of the requirements [Rupp06] available.

Level 4: Control oriented

At level 4 the standardised requirements support the success of the project by providing control parameter and test cases.

Requirements are not only qualitative but also quantitative verifiable.

There are catalogues of test cases available to support verification and validation

Level 5: Enhanceable

At level 5 the control oriented requirements are easy to change and in line with the business needs of the sponsor of the project and reflect the strategic decisions of the organisation.

4.2.2 Product Attributes

PA 1.1 Described Requirements

The described requirement is so clear an detailed that the estimation of the project planning and the further detailing and clarifying of requirements is possible.

The relevant stakeholders and the external interfaces are identified and documented.

Communication with stakeholders is initial planned and running.

Stakeholders contribute to the documentation of the requirements.

PA 2.1 Accurate Description

The accurate description attribute is a measure to what extent the requirements are fit for use.

The requirements are written to fulfil most of the criteria of IEEE 830

a) Correctness

b) Unambiguousness

c) Completeness

d) Consistency

e) Ranking of importance and stability:

f) Traceability

g) Understandability

h) The product attribute requires project internal rule for writing requirements.

i) The product attribute requires a quality gate or an exit criteria at the handover to the design phase.

PA 2.2 Acceptance Criteria

The acceptance criteria attribute is a measure to what extent the requirements support test and acceptance.

a) Acceptance criteria are verifiable

b) Acceptance criteria are suitable as a starting point for test case development

c) This product attribute requires project internal rules for formulating acceptance criteria .

PA 3.1 Standard Description

The standard description attribute is a measure to what extend the requirements catalogue is developed using an organization wide standard.

a) The standard must be written and proved to be suitable.

b) The standard reflects the recommendations of IEEE 830

c) The standard provides a content structure according to ISO 9126

d) The standard contains a guideline for requirement writing (not only criteria)

e) There are suitable templates and tools available to support the documentation of requirements according to the standard.

f) The standard is communicated to stakeholders and to all roles of the project which use the requirements as input for their work.

g) The usage of the standard is supported by project- and senior management.

h) The adherence of the requirements catalogue with the standard is verified.

PA 3.2 Quality Data

The quality data attribute is a measure to what extent the criteria of precise requirements are not only verified but also measured.

a) The quality of the requirements description is expressed by text based metrics

b) The manageability of the requirements catalogue is expressed by administrative metrics (see [Rupp06])

PA 4.1 Included Indicators

The included indicators attribute is a measure to what extent requirements are formulated in a quantitative way.

a) Requirements whose fulfilment can not only be verified but also measured are identified.

b) Measures for the fulfilment of single requirements or groups of requirements are defined.

c) the requirements are formulated using the defined measures and setting the goals for these measures.

PA 4.2 Test

The test attribute is a measure to what extent the requirements are detailed by test cases.

a) There is a test plan available with test steps and test objects

b) For each test step there i minimum one test object defined

c) Test cases are available for each test object

d) Requirements are mapped to test steps, test objects and test cases

Note: The attribute does not require formal methods for test case development

PA 5.1 Changeability

The changeability attribute is a measure to what extent the requirements catalogue or single requirements can be changed without raising inconsistencies.

a) Each requirement has one point of description in the requirements catalogue.

b) There is a standard procedure for the evaluation of the impacts of changes defined.

c) The history of each requirement can be stored.

d) Versions of requirements can be linked to change requests.

e) The complete adoption of each change request in the requirements catalogue can be verified.

f) Conflicts between change requests can be identified and solved.

PA 5.2 Constant Conformance

The constant conformance attribute is a measure to what extent a requirements catalogue fits the needs of the business during the whole project.

a) Each requirement is owned by an internal or external stakeholder.

b) There is a procedure in place to monitor the needs of internal or external stakeholders.

c) There is a procedure in place to monitor changes in the business strategy.

d) Requirements are linked to external reference documents (legal or technical standards etc.). There is a procedure in place to monitor changes in this documents and evaluate potential changes for the requirements catalogue.

e) For each requirement the probability of change is evaluated and documented.

4.2.3 Process preconditions

To make sure the requirements maturity is not only linked to product but also to process the model contains requirements for the process capability of the following key processes:

- Requirements elicitation
- System requirements analysis
- Software requirements analyses
- Change request management
- Configuration management
- Quality assurance
- Project management

The required capabilities are

For maturity level 1: All processes above at capability level 1.

For maturity level 2: All processes above at capability level 2.

For maturity level 3 or higher: All processes above at capability level 3.

4.3 The intended Use

As pre described, the requirements maturity model is derived to support management decisions at project and organization level.

Note that the model only indicates the necessity of decision making or risk management. It is not intended to provide a "one and only solution" Example: if a project has a requirements catalogue at Level 1 the management can't make a simple decision like lets improve the catalogue to level 2. To improve the requirements is a possible but not the only possible option. Given constraints might force the choose of other options or the cancellation of the project.

4.3.1 Evaluate risk

It seems to be evident that a requirements catalogue with maturity level 0 brings more risk into a project than a requirements catalogue with maturity level 5.

But also the improvement of a requirements catalogue might raise risks for the budget, the delivery date or for key resources to become bottlenecks.

4.3.2 Choose a procedure Model

As shown, one risk of project failure related to requirements is to choose the wrong procedure model.

The following table shows some recommendations based on the maturity of requirements

Level/approach	Agile	RuP	V-model	Waterfall
0	+	o	-	-
1	+	+	-	-
2	o	+	+	+
3	-	o	+	+
4	-	-	+	+
5	-	-	+	+

- not recommended (High risk or not efficient)

o might work dependent on constraints

+ recommended

4.3.3 Evaluate planning Parameters

The evaluation of the maturity of a requirements catalogue also allows to review the planning parameters.

As we know, estimates are derived from requirements by using formal methods and personal experience.

If the requirements catalogue has maturity level 0 it seems to be evident that the estimates are mostly based on the personal experience of the project leader. A significant variance between the estimates of different project leaders can be expected.

If the requirements catalogue is at level 5, estimates can be derived with formal estimation methods like COCOMO or function point. The variance of estimates is significant reduced.

4.3.4 Stop or Go decision

The measure of the maturity of requirements also supports stop or go decisions.

By defining the needed maturity level and evaluating the current maturity level the needed improvements may be estimated.

If the figures show a substantial risk for the business case an explicit stop or go decision should be made to make sure that the project has strong management commitment.

4.3.5 The benefits

Putting it all together the model provides substantial benefits for the project and organization management.

The management gets an easy to understand and unambiguous information about the quality of the requirements without the need to understand the content of the requirements, the quality assurance process or a set of metrics.

The model also delivers the necessary decision topics. That supports management commitment and project success.

5 Conclusion

Using instruments from the process capability area like the measurement framework of ISO 15504 is useful also in the product maturity area.

As shown with the requirement engineering and management as an example, crossing the border between process and product brings a chance of substantial benefits.

6 Further Work

The further work will have to address 3 topics:

Test the hypothesis with sufficient statistical data (instead of published reports and personal experience)

Test the hypothesis by using it for architecture, design, code and test

Test the hypothesis by evaluating if product aspects are useful to enhance the understanding of processes.

7 References

[Hood05] Colin Hood, Rupert Wiebel: Optimieren von Requirements Management & Engineering mit dem HOOD Capability Model Berlin 2005

[Standish 95-06] THE STANDISH GROUP REPORT CHAOS 1995 -2006

[Gilb04] Tom Gilb: Project Failure: Some Causes and Cures By Tom Gilb Web publishing 2004.

[Centreline04] 10 Major Causes of Project Failure Centreline Solutions Inc. Web publishing 2004

[Shin03] Yonghee Shin: A Maturity Model of Evaluating Requirements Specification Techniques. (Texas A&M University August 2003)

[Rupp06] M. Recknagel, C. Rupp: Messbare Qualität in Anforderungsdokumenten, manage it Vertikal, Ausgabe 2—2006 Sonderheft, S. 12-17, ap Verlag

[IEEE 830] IEEE Recommended Practice for Software Requirements Specifications. IEEE Standard 830-1998.

Using ISO/IEC 15504 to Validate a Set of Teamwork Factors

Antonia Mas, Esperança Amengual

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

Abstract. Software development process is a team activity and success in software projects depends largely on the performance of software teams. Therefore it is important to measure the effectiveness of software development teams. After establishing a set of teamwork factors for succeeding in software development projects which are expected to be the basis for the development of a teamwork assessment model, our current goal is to validate this set of factors against a software process assessment standard. In this article a mapping between the ISO/IEC 15504 best practices and the teamwork key factors is presented.

1 Introduction

As well as in team sports, success in software projects is conditioned to the good performance of software teams. When a project manager comes face to face with aggressive deadlines or has to make an important decision he really depends entirely on his team which is formed by different people, with different aspirations, interests and ways of working, and whose interpersonal relations determine the success or failure of the project.

Different investigations stand out on the importance of teamwork to succeed in software development projects. Tom De Marco and Timothy Lister, in their book *Peopleware: Productive Projects and Teams* [1], emphasise that software development is almost never a technical problem, but a human one. In the same way, other researchers agree on that human aspects of software development are more important than the technological aspects for better performance [2].

Teamwork has also been considered strategic for software process improvement. The Team Software Process^{SM} (TSP) developed by the Software Engineering Institute (SEI^{SM}) is "a framework and a process for building and guiding development teams" [3]. In the SEI^{SM} technical report [4] it is demonstrated that TSP can be used together with the Capability Maturity Model Integration to accelerate process improvement.

In accordance with the above mentioned authors, our investigation group has also considered good teamwork as an essential contribution to improve software processes. After taking part in an initiative on software process improvement in software small and medium enterprises [5, 6, 7], we have decided to open a new investigation path to study teamwork in software development.

The main goal of this article is to analyse the validity of a set of teamwork factors against the ISO/IEC 15504 process assessment standard. In section 2 the five identified teamwork key factors are summarized. In section 3, some ISO/IEC 15504 related aspects that we consider necessary for the understanding of the analysis are introduced. Finally, in section 4 the mapping between the teamwork factors and the best practices proposed by the standard is exposed.

2 Teamwork Key Factors

There is an agreement among teamwork investigations that the human aspects are a key point to control teams. Starting with this premise, different authors expose in their studies the elements that should be taken into consideration to efficiently work in a team. Coordination, communication, social interaction, role identification, leadership, and motivation are only some examples of the broad terminology used.

One of the first results of our investigation in teamwork has been the identification of a set of factors that must be considered for the success in software development teams. Thus, management, coordination, communication, composition and motivation have been established as the teamwork key factors that are expected to be the starting point to develop a teamwork assessment model for software projects [8]. This assessment model will define a set of teamwork best practices together with a rating scale to measure the efficiency of a software team. With this goal in mind, we first consider necessary to validate the identified teamwork key factors against a software process assessment standard. Since we are experienced in using the ISO/IEC 15504 standard for software process improvement [9] we have considered it as our reference framework to validate the set of teamwork key factors.

Table 1 summarizes the 5 teamwork key factors and the aspects that each factor includes. These aspects have been the starting point to find evidence of a teamwork factor in the ISO/IEC 15504 best practices.

Table 1. Teamwork key factors

Factor name	Aspects that the factor includes
Management	Team identity and common vision
	Definition of objectives and tasks
	Management resources and infrastructure
	Monitoring
Coordination	Identification of responsibilities and interactions among them
	Task and product relations
	Verification and control
Communication	Fluent communication
	Communication mechanisms
	Notification

Factor name	Aspects that the factor includes
Composition	Role definition
	Personnel selection
	Responsibilities assignation
	Training
Motivation	Commitment
	Competencies and expectations
	Recognition mechanisms

3 The ISO/IEC 15504 Best Practices

ISO/IEC 15504 is an international standard for process assessment and improvement.

Part 5 of the standard [10] provides an exemplar Process Assessment Model that meets the requirements of the normative part of the standard [11] and that supports the performance of an assessment by providing indicators for guidance on the interpretation of the process purposes and outcomes as defined in ISO/IEC 12207 Amd 1 and Amd 2 [12, 13].

The Process Assessment Model (see Figure 1) is based on the principle that the capability of a process can be assessed by demonstrating the achievement of process attributes on the basis of evidences related to assessment indicators.

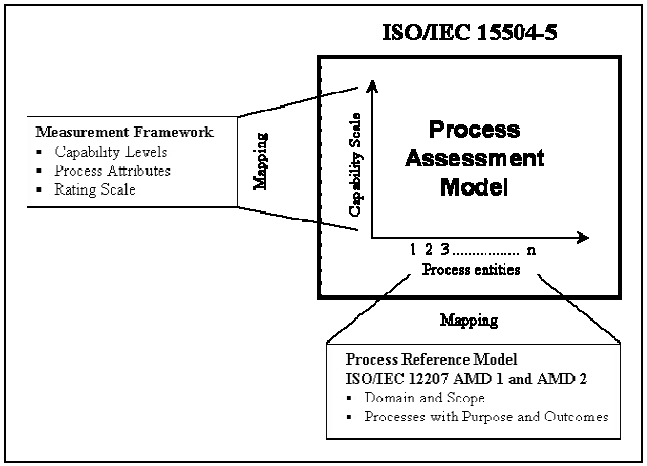


Fig. 1. ISO/IEC 15504 Process Assessment Model relationships

There are two types of assessment indicators: process capability indicators and process performance indicators. As our goal is to look for evidence of teamwork in

the processes best practices, we have focused on the process performance indicators that relate to individual processes to explicitly address the achievement of the defined process purpose. More specifically, we have analyzed the set of ISO/IEC 15504 Base Practices for each process that provide a definition of the tasks and activities needed to accomplish the process purpose and fulfil the process outcomes.

4 A mapping between the teamwork key factors and the ISO/IEC 15504 Base Practices

This section shows the connection between the teamwork key factors and the ISO/IEC 15504 Base Practice indicators. As the complete mapping is too extensive to be detailed in this article, only a meaningful part of it is exposed. Firstly, as an example of a particular factor, the results for the Composition factor are detailed. Secondly, the mapping between the five teamwork key factors and the process group that considers the greatest number of teamwork aspects, the Management Process Group (MAN), is exposed.

4.1 Mapping the Composition factor

As well as for the whole set of factors, to analyse the relation of the Base Practices with the Composition factor in particular, it first has been necessary to detail the different aspects that compose this factor. These aspects have been the basis to look for related ISO/IEC 15504 Base Practices. The detail of each aspect that form part of the Composition factor is shown in table 2 and the selected Base Practice indicators for this factor are detailed in table 3.

Table 2. Composition aspects

COMPOSITION FACTOR		
Aspect		**Meaning**
1	Role definition	Identify and define the roles that can be assigned to the different team members. Define the technical, management and collaboration skills necessary to perform each role.
2	Personnel selection	Identify and select the most suitable and competent personnel for each team role.
3	Responsibilities assignation	Assign responsibilities and authorities to the selected members. Each member needs to understand the tasks and responsibilities of his/her role as a team member.
4	Training	Actions oriented to identify the skills that have to be provided by training programs.

Table 3. Base Practice indicators identified for the Composition factor

COMPOSITION FACTOR		
Base Practice	**Meaning**	**Related aspect**
MAN.3.BP6	Identify the experience, knowledge and skill requirements of the project and apply them to the selection of individuals and teams.	2
MAN.3.BP9	Allocate specific responsibilities and ensure that the commitments are understood and accepted.	3
MAN.4.BP2	Develop an overall strategy including necessary resources and responsibilities to achieve the defined goals	3
RIN.1.BP1	Identify and evaluate skills and competencies needed by the organization to achieve its goals.	1
RIN.1.BP2	Define objective criteria that can be used to evaluate candidates and assess staff performance.	2
RIN.1.BP3	Establish a systematic program for recruitment of staff competent to meet the needs of the organization.	2
RIN.1.BP4	Define and provide opportunities for development of the skills and competencies of the staff.	4
RIN.1.BP5	Define the structure and operating rules under which projects operate.	3
RIN.1.BP10	Maintain adequate records of staff, including not only personal details, but also information on skills, training completed, and performance evaluations.	2
RIN.2.BP1	Develop a strategy for training including how the training needs will be identified, how the needed training will be developed or acquired, and how the training will be performed.	4
RIN.2.BP2	Identify and evaluate skills and competencies to be provided or improved trough training.	4
RIN.2.BP3	Develop or acquire training that addresses the common training needs.	4
RIN.2.BP4	Identify and prepare the execution of training sessions, including the availability of the training materials and the availability of personnel to be trained.	4
RIN.2.BP5	Train personnel to have the knowledge and skills needed to perform their roles.	4

COMPOSITION FACTOR		
Base Practice	**Meaning**	**Related aspect**
RIN.2.BP6	Maintain adequate records of the training completed by the staff.	4
SUP.4.BP1	Identify the participants of management and technical reviews.	2
SUP.5.BP1	Define the audit team.	2
SUP.5.BP2	Select independent, impartial and objective auditors.	2

As it can be observed in the table, the four particular aspects that the Composition factor includes have been mapped to at least one Base Practice. Therefore, it can be stated that this factor is fully contemplated in the standard. Despite this, the different aspects are not considered by the standard in the same proportion. As it is shown in the following pie chart, training and personnel selection are related to a greater number of ISO/IEC 15504 best practices than responsibilities assignation or, the less connected aspect, the role definition.

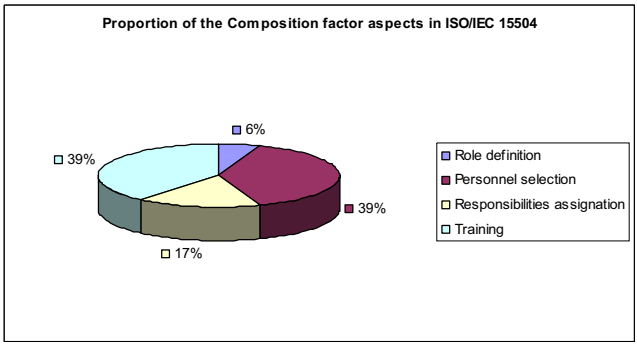


Fig. 2. Proportion of the Composition factor aspects in ISO/IEC 15504

Regarding to distribution of the Composition factor among processes, as it can be observed in table 3, this factor has been covered by only three process groups: the Management Process Group (MAN), the Resources and Infrastructure Process Group (RIN) and the Support Process Group (SUP).

4.2 Mapping the teamwork key factors in the Management Process Group (MAN)

The processes of the Management Process Group (MAN) contain practices to manage any type of project within a software life cycle. As table 4 demonstrates, there is a significant connection between the teamwork factors and the Base Practices of these processes.

Table 4. Base Practice Indicators identified in the Management Process Group

Process	TEAMWORK KEY FACTOR				
	Management	**Composition**	**Communication**	**Coordination**	**Motivation**
MAN.1	MAN.1.BP1 MAN.1.BP6		MAN.1.BP5	MAN.1.BP1	MAN.1.BP4 MAN.1.BP6 MAN.1.BP7
MAN.2	MAN.2.BP1 MAN.2.BP2 MAN.2.BP5 MAN.2.BP6	MAN.2.BP1 MAN.2.BP2	MAN.2.BP1 MAN.2.BP2		MAN.2.BP6
MAN.3	MAN.3.BP1 MAN.3.BP3 MAN.3.BP4 MAN.3.BP5 MAN.3.BP7 MAN.3.BP10 MAN.3.BP11 MAN.3.BP12 MAN.3.BP13	MAN.3.BP6 MAN.3.BP9	MAN.3.BP10 MAN.3.BP11 MAN.3.BP13	MAN.3.BP5 MAN.3.BP7 MAN.3.BP8 MAN.3.BP10 MAN.3.BP11	
MAN.4	MAN.4.BP1 MAN.4.BP2 MAN.4.BP8	MAN.4.BP2	MAN.4.BP7		
MAN.5					
MAN.6	MAN.6.BP1 MAN.6.BP9		MAN.6.BP1 MAN.6.BP8 MAN.6.BP9	MAN.6.BP8 MAN.6.BP9	

The following bar chart shows the percentage of selected Best Practices per teamwork factor in the processes that constitute the Management Process Group.

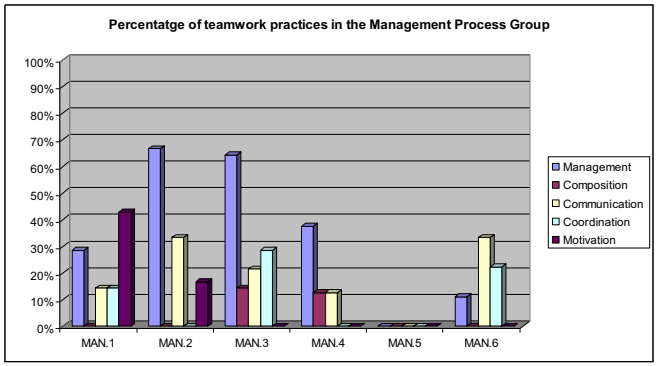


Fig. 3. Percentage of teamwork practices in the Management Process Group

As this bar chart reveals, the five teamwork factors are addressed by the processes of the Management Process Group. In particular, there are evident connections between the Base Practices of these processes and the Management factor. More concretely, this factor covers more than 60% of the total number of ISO/IEC 15504 best practices that the standard proposes for the Organization management process (MAN.2) and the Project management process (MAN.3).

6 Conclusions and Further Work

The results of the mapping between the five teamwork fey factors and the ISO/IEC 15504 best practices introduced in this article demonstrate that the whole set of factors is addressed by the standard in different proportion through the different process groups.

The main goal of the mapping was to validate the initial set of teamwork factors against a recognized software process standard. Therefore, the evidence of each particular factor in the standard lays the foundations for this set of factors to be considered the basis for a future teamwork assessment model. Moreover, the particular ISO/IEC 15504 Base practices that have been related to teamwork can also be a starting point to infer teamwork best practices.

The proportion of selected Base Practices per process group is another interesting result because, as it can be observed in the following pie chart, teamwork practices are concentrated in the Management Process Group (MAN) and in the Resources and Infrastructure Process Group (RIN). Since our long-term objective is to validate if teamwork improvement can result in software process improvement, focusing on this

two process groups to identify teamwork best practices could be a good beginning for the development of the teamwork assessment model.

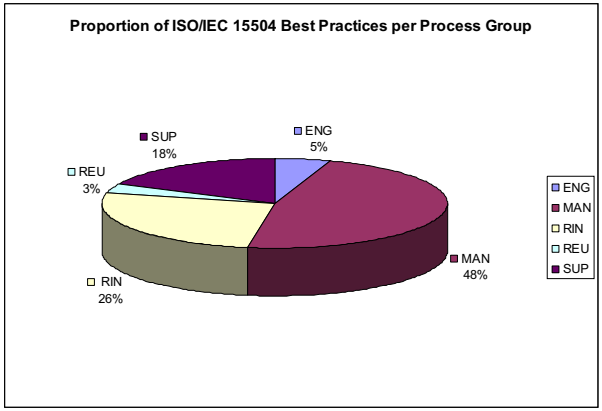


Fig. 4. Proportion of ISO/IEC 15504 Best Practices per Process Group

Finally, to demonstrate the importance of teamwork, our intention is to apply the teamwork assessment model in parallel with a process assessment against the requirements of the ISO/IEC 15504 standard

Acknowledgements. This investigation has been supported by SOAQTest: Calidad en los procesos de desarrollo y pruebas en arquitecturas orientadas a servicios (TIN2007-67843-C06-04).

References

[1] De Marco, T.; Lister, T.: “Peopleware: Productive Projects and Teams”, Dorset House Publishing Company, 2nd edition (1999)

[2] Gorla, N., Wah Lam, Y. "Who Should Work with Whom? Building Effective Software Project Teams". Communications of the ACM, vol. 47, no. 6, (2004), pp. 79-82

[3] Humprey, Watts S. TSP^{SM} - Coaching Development Teams. Addison-Wesley, United States (2006)

[4] CMU/SEI-2004-TR-014. Mapping TSP to CMMI. Software Engineering Institute (2005)

[5] Amengual, E.; Mas A. "A New Method of ISO/IEC TR 15504 and ISO 9001:2000 Simultaneous Application on Software SMEs". Proceedings of the Joint ESA - 3rd International SPICE Conference on Process Assessment and Improvement, (March 2003), pp. 87-92

[6] Mas, A.; Amengual, E. "ISO/IEC 15504 Adaptation for Software Process Assessment in SMEs". Proceedings of the International Conference on Software Engineering Research and Practice, (June 2003), pp. 693-697

[7] Mas, A.; Amengual, E. "A Method for the Implementation of a Quality Management System in Software SMEs". Software Quality Management XII. New Approaches to Software Quality, The British Computer Society, Great Britain (2004)

[8] Amengual, E.; Mas, A.: "Software Process Improvement through Teamwork Management". In: Münch, J., Abrahamsson, P. (Eds.) Profes 2007. LNCS, vol. 4589, pp. 108-117. Springer, Heidelberg (2007)

[9] 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

[10] ISO/IEC 15504-5:2006. Information technology -- Process Assessment -- Part 5: An exemplar Process Assessment Model. International Organization for Standardization (2006)

[11] ISO/IEC 15504-2:2003. Information technology -- Process assessment -- Part 2: Performing an assessment. International Organization for Standardization (2003)

[12] ISO/IEC 12207:1995/Amd 1:2002 Information technology -- Software life cycle processes. International Organisation for Standardization (2002)

[13] ISO/IEC 12207:1995/Amd 2:2004 Information technology -- Software life cycle processes. International Organisation for Standardization (2004)

The role of usability in the web development process – an analysis of SMEs providing MIS applications for the web

Rory V O'Connor and Bairbre Baxter
Dublin City University
Ireland

Abstract

The number, variety and complexity of Web applications is growing steadily and a large number of these applications are developed by Small and Medium Enterprises (SMEs). This growth creates a need for supporting development processes and usability standards that meet the unique needs of web application development. This study analyses the practices of several SMEs who develop web applications through a series of case studies. With the focus on SMEs who develop web applications as Management Information Systems and not E-Commerce sites, informational sites, online communities or web portals. This study gathered data about the usability techniques practiced by these companies and their awareness of usability in the context of the software process in those SMEs. The contribution of this study is to further the understanding of the current role of usability within the software development processes of SMEs who develop web applications.

Keywords

Software process improvement, Software process, Usability, Web development, SME

1 Introduction

Since the introduction of the Internet, web applications have moved beyond information sharing to a point where most traditional standalone applications have a web-enabled version [1]. Today the term web applications represent anything from information portals to online communities. This study focuses on web applications as Management Information Systems (MIS) accessed via a web browser with a central database backend. It focuses on the following definition of a web application proposed by [2]: "*These new web applications blend navigation and browsing capabilities, common to hypermedia, with 'classical' operations (or transactions), common to traditional information systems*". This study does not consider in its scope E-Commerce sites, informational sites, online communities or web portals.

With the growth of the software industry, many development process models have emerged, such as the waterfall, iterative and agile models. Companies are also placing an increasing emphasis on the importance of compliance with standards such as ISO 9001 or the use of best practice models such as the Capability Maturity Model Integration (CMMI). But despite the number and variety of models and frameworks, there is evidence that SMEs find it difficult to adhere fully to any one model or set of standards [3].

Recently there has been a call for new development process models that address the unique requirements of web application development [4]. Such requirements include a short development lifecycle and a shorter shelf life of new functionality. They must also keep pace with the rapidly changing technology on which they rely. There are general guidelines available on what a web application process should incorporate. Suggestions include combining the activities of traditional models with those of hypermedia design models [5]. Alternatively, an incremental process is recommended, incorporating activities that address the needs of web application development [6]. Despite these guidelines, there is evidence that most web development is still largely ad-hoc and researchers liken it to the early days of traditional software development [7, 8]

1.1 Research Aims and Objectives

This study examines SMEs understanding of usability, what usability techniques they currently practice and how well they believe usability is represented in their development process. It analyses the software development process SMEs claim to use and looks at whether the process is actually followed in a typical project. By comparing results across several case study companies, this study investigates whether common issues and attitudes exist and how their practices compare to software development models and usability standards. By investigating the typical development process and what usability techniques are being used, the aim of this study is to set the groundwork for further investigation into whether SMEs find it difficult to follow software development process models and UCD models when developing web applications. Accordingly, the objectives of this study are to:

- Explore the software development processes in practice by SMEs who develop web applications.
- Investigate the SMEs understanding of usability and assess their level of commitment to it within the development process.
- Investigate the gap between the development processes practiced by SMEs developing web applications and the proposed software development process models, standards and best practices.
- Investigate the gap between usability awareness and practices among SMEs and usability standards and UCD guidelines.
- Gain an understanding of why SMEs do, or do not, integrate usability into their web development process.

2 Usability and Web Development Processes

Although usability is gaining widespread recognition, confusion exists as to what is meant by the term usability [9]. For some it focuses on the User Interface, dealing with issues such as user of colour, pleasing layout and consistent terminology. For others it deals with the software's overall structure, how productively it allows the user to complete their tasks and how easy it is to learn [10]. This study adopts the definitions put forward by the ISO (ISO 9241-11) which defines usability as: "*the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use*".

The process by which one achieves good usability in a product is known as User-Centred Design (UCD). This is also referred to as usability engineering or human-centred design. Many UCD design models put forward and all contain the key element of involving the user in all stages of the development process [11]. This is in contrast to a traditional software development process, which only involves the user in specific stages of the lifecycle, such as requirements analysis and acceptance testing.

Studies have shown that user-centred design techniques are still underused among development teams [12] and most usability issues are only detected during testing and after deployment [13, 14]. Of those practicing UCD, one investigation revealed that the majority of methods in practice were informal, low-cost user-centered design methods. The most commonly used methods were iterative design, usability evaluation, task analysis, informal expert review, and field studies [15]. Obstacles given for not implementing UCD techniques include a lack of awareness of usability across the company, lack of usability experience, poor management support and marketing pressures [16]. Another reason given is the fact that UCD techniques are developed in isolation from the software engineering community and real company environments and thus do not take into account how well they will work in terms of team buy-in, and resources [17].

2.1 Web Development Processes

Many current software development models have been criticized as not meeting the unique requirements of web application development [7, 4] and accordingly there is a need to develop new models that address the needs of web application development [4]. The absence of a well-defined model for web applications has been explained by two causes. Firstly, the scope of how a web application is defined varies greatly. Secondly, the web's legacy is as an information platform rather than an application platform [8].

There are some general guidelines available on creating a development process for web applications. [5] suggests combining the activities of traditional lifecycles with those suggested for hypermedia. [6] suggests an evolutionary, or incremental, process which addresses the needs of web application development through the following activities: formulation, planning, analysis, modelling, page generation, testing and customer evaluation. Finally, many agree that regardless of the type of application being developed, the basic principles of software engineering should always apply. Good design, solid testing and change control should all be used as they are historically proven to work. [18, 6].

2.2 Role of Usability in Web Development Process

Web application usability goes beyond interface design and interaction issues specific to web pages. This study has found that research on usability standards for the web focuses almost exclusively on web sites and there is a lack of usability standards for web applications and developers admit to defining standards as they go. They also express confusion as to what standards they should conform to, those for web sites or traditional applications [19]. In the absence of clear recommendations, this study looks at how web applications share characteristics of both traditional applications and web sites.

Web application front ends are accessed via a browser, just as web sites are. As far as usability for

the user interface is concerned, web applications can borrow from guidelines common to web sites. Web applications share other usability issues with web sites, such as: download times, browser preferences and access via different devices, such as PDAs [11]. On the other hand, web applications may differ from web sites when it comes to the importance of learnability. Learnability may be less critical in web applications compared to web sites as they are likely to be accessed on a more frequent basis. There is also a greater chance that some degree of training or documentation is available for web applications compared to informational web sites [20].

There is little evidence available on the level of usability being delivered in real web applications today and how today's end users feel about usability standards. This may be put down to the reluctance of companies to allow such information to become public. But usability concerns for web sites focus on the UI and interaction issues dealing with information, such as searching.

3 Case Studies

The case studies were restricted to companies who develop web applications. The definition of web applications presented in section 1 has formed the basis for selecting suitable companies. It was not limited to any genre of web application or to a geographical area. It was also considered immaterial if a company also developed traditional applications as long as a significant portion of development efforts focused on web application development. The primary source for identifying case studies was through the researchers contacts, with possible companies being assessed Through this process, five suitable case studies participants were identified, as listed in Table 1. Appropriate interviewees were then selected within these organizations. The interviewees were all IT staff who were directly involved in web application development with the company. The job titles of those interviewed included Web Development Manager, Product Manager and Software Development Manager among others.

Company	No. Developers	Market Sector	Interviewee(s)
A	20	GIS Systems	Senior Developer & Interaction Designer
B	20	Finance	Technical Architect
C	17	Education	Development Manager
D	36	Emergency Response	IT Manager
E	15	Education	Project Manager

Table 1: Case Study Companies

An interview guide was prepared for use in the semi-structured interviews which comprised both factual questions and open-ended questions designed to explore the interviewee's attitudes and opinions. It was designed to be semi-structured based on the assumption that additional questions would be asked depending on the direction in which the answers went. The guide was deigned so that each interview would be completed within an hour, in order to ensure that interviewees would not lose focus. The five main topic area covered by the interview guide were:

- General background information about the company and its business sector.
- The organization's software development process and its practice.
- The organization's understanding and awareness of usability.
- Usability Practices: Usability activities within the development process.
- The interviewee's opinion of usability in relation to the company's products.

Detailed notes were taken during each interview and any additional questions that were asked were also noted. Each interview was also recorded on tape. After each interview, the tape recordings were transcribed and the interview notes were reviewed and documented. This material was then used as the basis for within-case analysis. The researchers looked for interesting findings or contradictory answers and wrote a summary of observations for each case. All five interviews took place over a two-month period. After all of the interviews had been completed, the researchers began within-case analysis. After the within-case analysis was complete, cross-case analysis was carried out,

4 Analysis

This section presents the cross-case analysis of the data collected during the case study interviews. It examines the findings of the interviews under the areas of Software Process, Usability Awareness, Usability Practices and Product Usability. Firstly, it looks at the software practices followed by the case study companies and compares them to recommended practices as discussed in the literature and whether they have adopted suggested practices for web application development. It then discusses the awareness of usability and investigate usability practices of the case study companies and examines the gap between their practices and suggested usability design techniques. Lastly, it discusses the opinions of the interviewees about the usability of their products and examines the lack of evidence available on the level of usability of today's web applications.

4.1 Software Process

Of the five case studies, two companies use RUP as their development method, one uses an Agile approach and the other two use an internally-developed process based on a waterfall style model. Only the two companies using RUP had a fully documented process. The company using an Agile approach had a partially documented process and the two companies using an internally-developed process had not documented it at all. Analysis of the development process revealed that all five companies were knowledgeable and clear in describing the steps that they follow, regardless of whether it was documented or not. All but one of the companies believed the process was being followed in all projects. However, four out of five companies also cited deviations from the process.

An interesting finding was that three of the companies had recently undergone significant improvements to their processes. One company had hired a project manager with the responsibility of establishing a more structured, repeatable development process. Another set up a new test team and formalized the build process. It was evident that these companies were moving in the right direction while still being aware that they had more improvements to make.

4.1.1 Gap in Software Process

None of the companies were following any of the available development models without having customised it to their needs. When describing their development process, all five companies reported having a Requirements Analysis phase at the beginning of the lifecycle. Much of the literature cites poor requirements as the cause of many subsequent problems in the software. But [21] believe that in web projects, clients do not have a clear enough understanding of their requirements at the beginning of a project for existing software processes to be effective. They believe that web development companies should adopt an iterative approach that incorporates client-developer interaction and that assesses partial designs in order to clarify the client's requirements. Although only one company cited poor Requirements Analysis as a problem in their process, there appears to be a lack of awareness that a key advantage of the iterative design process is its ability to involve the end user early in the product lifecycle. Of the three companies following an iterative process, only two delivered interim software builds to the client. But both of these companies described the client as a distinct entity to the end user of the system. Delivery of the builds appeared to be more to meet the contract deliverable rather than a design tool.

The literature suggested that web application development can be likened to the early days of traditional software development, when applications were mostly being developed in an ad-hoc manner. But this study has revealed that all five case studies have a defined development process. Although the process may not have been documented in two cases, all of the companies were able to clearly describe the steps involved in their process and believed it to be a clearly-defined, repeatable process. They were also able to acknowledge deviations from the defined process. These findings suggest that although there appears to be a need for a process suitable to small companies developing web applications, practices are more formal than anecdotal evidence suggests.

4.2 Usability Awareness

All of the companies had very little awareness of usability standards, with only one company having a good knowledge of usability. Most of the companies believed usability was well represented in their development process and that usability awareness was good throughout the company. It emerged that two companies had a limited understanding of usability awareness, citing look and feel as the primary element. The other three companies had a deeper understanding, describing usability as the need to support the user tasks. An interesting finding was that those companies that showed a deeper understanding of usability were also the ones doing business on a tender basis. It is possible that in order to win tenders, companies must ensure that they respond to the client's needs. It is also possible that during the development process, the client has much deeper involvement compared to those companies who are selling their application on an off-the-shelf basis.

Analysis of users needs showed that the most commonly reported need was intuitive use. Two companies remarked that having to do as few clicks as possible was important for their users, while another phrased this as fast use. Other needs cited were easy navigation, quality of information and responsiveness. One company observed that their users simply like what they are used to. This is an interesting challenge when developing web applications because it is possible that users are used to desktop applications but have less experience with web applications. This is reflected in the fact that one company said that their biggest challenge was delivering more and more complex functionality via the web and still trying to maintain a high level of usability. The challenge is to develop a web application that delivers a high level of ease of use and learnability so that it becomes irrelevant to users that they achieve their goal in a slightly different way to before. The researchers also believes that novice users may benefit greatly from education from the development company on the advantages the web brings before assuming that the client wants a mirror image of the desktop application functionality.

Only one company reported that awareness of the user needs and their IT skills was poor. They acknowledged that this was reflected in the fact that they were still delivering new functionality with poor usability. Most of the companies felt that awareness among staff of the client needs grows with the experience of working on a project and through good requirement specifications.

4.2.1 Gap in Usability Awareness

Analysis of how the interviewees defined usability supports the evidence that confusion still exists as to what is meant by usability. For some usability refers to the UI and for others it means how productively the system allows users to complete their task. Two companies defined usability in terms of the UI and the other three defined it in terms of supporting the user's task. It is encouraging that three companies defined usability as the extent to which it supported the user tasks. But only one company mentioned efficiency as an element of usability. This is particularly interesting in terms of web applications because efficiency has been cited as one of the most important aspects of usability for the web. Also, none of the companies remarked on effectiveness or satisfaction as key elements of usabilityMost of the companies have reached an understanding that a system should enable a user to reach his goal but they lack the awareness of the fact that it should enable them to do so in as productive and pleasing a manner possible.

Rather than dismissing those who defined usability primarily in terms of look and feel as having a poor understanding of usability, it is worth looking at the fact that most of the companies did not mention look and feel at all. Although industry definitions make it clear that usability is much more about the look of a product, [5] cites the 'degree of visual quality' as a key element of usability for web applications. This finding supports the observations by [19] who noted that developers are confused about whether they should conform to web site or traditional application standards. It is encouraging that three of the companies described usability in terms of reaching user goals but the importance of look and feel for web applications cannot be dismissed. This raises the need for a clearer definition of usability for web applications, one that embraces the need to support the user goals yet recognizes the visual elements web applications share with web sites.

Analysis of how the companies described the usability needs of their user shows a contradiction with their definitions of usability. For example, when describing their understanding of usability, no companies mentioned efficiency or productive use. But when discussing the needs of their user, two

cited the most important element as efficient use of the product. Another example is that although two companies defined usability in terms of look and feel, none regarded it as a usability need for their users. Yet most companies recognised it as a key element in attracting new customers. The most common usability needs cited centred around ease of use, although it was described in different ways. One company described it as learnability, another as ease of use and two as intuitiveness. This is interesting when compared to claims by [20] who suggested that learnability is not as important in web applications compared to web sites because the user would be more likely to have undergone training or have documentation available.

4.3 Usability Practices

Only two of the five companies had internal staff dedicated to usability design practices and one of these was a part-time employee working from home. A third company used external consultants to conduct usability evaluations of their product during its initial development. Three of the five companies gathered usability requirements as part of requirements analysis. In two of these companies, they do not explicitly refer to them as usability requirements, rather they were gathered as part of the general task requirements for the user. These were the same companies that defined usability in terms of supporting the user's tasks. It is difficult to see how the user can explicitly provide all of their usability requirements without ever referring to them as such.

In terms of the overall product design, three of the companies had a formally-established software design team in place and the other two had lead architects responsible for product design. They were responsible for the overall vision and direction of the product. It is of concern that there was no mention of usability being represented at this level of design. It appears that usability tasks are being practiced at grassroots level and are of less concern during the high level design of products. This suggests that usability is not a concern at the upper management level yet management support is critical for it to grow in importance. Although all five companies considered themselves to be offering a good level of usability, only one of the four companies had a management-driven approach to practicing usability techniques.

Two of the five companies claimed to do usability testing, with one reporting that that this was done as part of Acceptance Testing. The researchers believes that there is a lack of understanding as to what usability testing is and it is confused with User Acceptance Testing. Two companies required that the client must sign off on the product based on acceptance testing. This is a positive step although not an efficient means in catching usability issues at the end of the project lifecycle.

4.3.1 Gap in Usability Practices

When asked who was responsible for usability in the end product, two companies cited the client. This is interesting considering the fact that these companies never explicitly discuss usability with the client, so it is difficult to see to what degree they are responsible. Although all companies demonstrated a degree of collaboration with the client during Requirements Analysis, only one company sought approval from the client on the final set of requirements. The most interesting observation was that none of the companies openly discussed usability requirements with their clients but incorporated it into the task requirements. This suggests that companies expect their clients to be able to represent their usability needs without having explicitly referred to usability.

The lack of UCD practices was apparent across all of the case studies, regardless of whether they developed bespoke applications or software for sale to multiple customers. The findings revealed that the three companies developing bespoke software were the only ones who claimed to gather usability requirements. However, the evidence on overall usability practices in this sample size did not suggest that the nature of applications being developed had any bearing on the level of UCD techniques being practiced.

Analysis of the development process has shown that three of the companies are following an iterative process, which is encouraged by UCD experts as a critical factor in ensuring good usability in the end product. But during their iterative design phase, only two companies provide early prototypes to the clients for analysis. Evidence shows that finding usability issues at the end of a project life cycle is the most inefficient way to resolve them. For this reason, it is worrying that most of the companies are not

involving their users from the early stages of the design process. It appears that between Requirements Analysis and Acceptance Testing, there is very little interaction between the client and the development team.

It should also be noted that there was almost no distinction in any company between client and end user. One company noted that the client might review the requirements despite the fact that they are not necessarily knowledgeable about the end user's needs. It was clear that these companies recognised the fact that they had to please the client first and foremost. But this assumes that the client will represent the end users needs and if the end user is not happy with the end product, it is unlikely that the client will take responsibility.

The evidence suggests that meeting usability needs is considered by companies to be a part of good functional and U.I design, rather than a set of independent tasks. These companies have not adopted specific usability techniques in their development process. This supports the evidence that UCD techniques as criticized as unsuitable due to the fact that they were developed outside the field of software development. Despite not using usability techniques, most of these companies demonstrated a belief that they are supporting the usability needs of the user through good task analysis. [21] believe that web-based applications place increased emphasis on user interactions. It suggests that the nature of web applications means that there is already more focus on the user experience compared to developing traditional applications.

4.4 Product Usability

All of the companies believed that usability was very important for attracting new customers. They unanimously claimed that the usability of their product was very good. However, it was outside the scope of this study to examine the usability of the products developed by the case study companies. For this reason, it was not possible to verify the claims made by the interviewees about the usability of their products. All five companies claimed that the usability of their product was better than the competition, another claim which could not be verified without assessing the usability of their products and their competitor's products.

4.4.1 Gap in Product Usability

This study found no evidence on the level of usability being delivered in web applications today. This has been justified by the fact that companies would naturally be reluctant to reveal negative feedback about their web applications. Accordingly, it was not possible to compare the opinions about the usability other companies products with those of the case study companies. As previously stated, this study also did not review the usability of the products developed by the case study companies as it was considered outside its scope. For this reason, it was not possible to compare the usability of the case studies products against those of other companies.

5 Discussion

The cross-case analysis has revealed differences between current practices among SMEs and industry standards for software development processes and usability practices. The key gaps between these standards and current practices are outlined below:

- SMEs are not using a development process designed to meet the specific needs of web application development.
- There is little use of UCD techniques in the development process:
    - Usability requirements are not gathered independently.
    - No formal usability testing.
    - No involvement of end user in design process.
    - Little practice of usability evaluations.

- The SMEs definition of usability is limited and inconsistent.
- There is a need for a definition of usability specifically for web applications.
- Uptake of, and interest in, best practice frameworks is poor.
- There is a need for open discussion with clients and end users on usability requirements.
- There is little awareness of usability standards and they are considered too vague to implement in real projects.
- Few staff members with UCD experience.

Other findings of less critical importance were:

- The definitions of usability made no provision for 'quality in use', such as satisfaction or efficiency.
- No usability representation during high level design of products.
- Descriptions of usability contradicted their awareness of the end user's usability needs.
- Regardless of the process model, interviewees demonstrated a good understanding of their process and acknowledged deviations.
- SMEs were positive in the direction they were taking through recent efforts to improve their process.

5.1 Conclusions

The findings show interesting similarities with our background literature review, which revealed that there were no proven process models available that met the specific needs of web development. This study showed that none of the companies were using a development process designed specifically for web application development. It also supported evidence that the use of best practice frameworks has been particularly slow among SMEs.

The literature also suggested that the practice of UCD techniques was slow, which was corroborated with the evidence from these case studies. The findings also uphold suggestions that web developers are confused about how to implement usability. Analysis of the interviews showed that the definitions of usability were inconsistent and that there is still is a need for a definition of usability specifically for web applications. There was also very little awareness of usability standards. Also of concern is the lack of involvement of end users in the development process.

There were positive findings in that the companies were demonstrating recent improvements in their process and an acknowledgement of process shortcomings. Interviewees demonstrated a good understanding of their process, regardless of whether it was documented or not. There was also a unanimously high level of pride in the end product.

6 References

[1] Ginige, A. and Murugesan, S., "Web Engineering: An Introduction", IEEE Multimedia, Jan-Mar 2001.

[2] Baresi, L., Garzotto, F., Paolini, P., "From Web Sites to Web Applications: New Issues for Conceptual Modeling", Lecture Notes in Computer Science, Vol. 1921, 2000.

[3] Coleman G. and O'Connor R.,"Software Process in Practice: A Grounded Theory of the Irish Software Industry", EuroSPI, LNCS 4257, Springer-Verlag, 2006, pp. 28-39

[4] Ginige, A. and Murugesan, S., "The Essence of Web Engineering", IEEE Multimedia, Apr-Jun 2001.

[5] Fraternali, P., "Tools and Approaches for Developing Data-Intensive Web Applications: A Survey", ACM Computer Surveys. Vol. 31, No. 3, 1999.

[6] Pressman, R., "What a Tangled Web We Weave", IEEE Software, Jan-Feb 2000.

[7] Avison, D. and Fitzgerald, G., "Where now for Development Methodologies", Communications of the ACM, Vol. 46, No. 1, 2003.

[8] Gellersen, H. and Gaedke, M., "Object-Oriented Web Application Development", IEEE Internet Computing, Jan-Feb 1999.

[9] Frokjar, E., Hertzum, M., Hornbak, K., "Measuring Usability: Are Effectiveness, Efficiency, and Satisfaction Really Correlated?", Proceedings of the SIGCHI conference on Human factors in computing systems, ACM Press, Apr 2000.

[10] Juristo, N., Windl, H., Constantine, L., "Introducing Usability", IEEE Software, Jan-Feb 2001.

[11] Mayhew, D., "The Usability Engineering Lifecycle", Morgan Kaufmann, 1999.

[12] Seffah, A. and Metzker, E., "The Obstacles and Myths of Usability and Software Engineering", Communications of the ACM, Vol. 47, No.12, 2004.

[13] Folmer, E., van Gurp, J., Bosch, J., "A Framework for Capturing the Relationship between Usability and Software Architecture", Software Process Improvement and Practice, Vol. 8, Issue 2, 2003.

[14] Anderson, J., Fleek, F., Garrity, K., Drake, F., "Integrating Usability Techniques into Software Development", IEEE Software, Jan-Feb 2001.

[15] Mao, J., Vredenburg, K., Smith, P., Carey, T., "The State of User-Centered Design Practice", Communications of the ACM, Vol. 48, No. 3, 2005.

[16] Radle, K. and Young, S., "Partnering Usability with Development: How Three Organizations Succeeded", IEEE Software, Jan-Feb 2001.

[17] Wixon, D., "Evaluation Usability Methods: Why the Current Literature Fails the Practitioner", Interactions, Jul-Aug 2003.

[18] Constantine, L. and Lockwood, L., "Usage-Centered Engineering for Web Applications", IEEE Software, Mar-Apr 2002.

[19] Cloyd, M., "Designing User-Centered Web Applications in Web Time", IEEE Software, Jan-Feb 2001.

[20] Lazar, J, "User-Centred Web Development", Jones and Bartlett Computer Science, 2000.

[21] Lowe, D. and Eklund, J., "Client Needs and the Design Process in Web Projects", Journal of Mobile Multimedia, Vol. 1, No. 1, 2005.

7 Author CVs

Rory V O'Connor

Dr. Rory V O'Connor is a senior lecturer in the School of Computing at Dublin City University and a researcher with Lero (The Irish Software Engineering Research Centre). He received a PhD. in Computer Science from City University (London) and an M.Sc. in Computer Applications from Dublin City University. He has previously held research positions at both the National Centre for Software Engineering and the Centre for Teaching Computing, and has also worked as a software engineer and consultant for several Irish and European technology organisations. His research interests are centred on the processes whereby software intensive systems are designed, implemented and managed, in particular, methods, techniques and tools for supporting the work of software project managers and software developers in relation to software process improvement, software project planning and management of software development projects. He can be contacted by email at: roconnor@computing.dcu.ie

Bairbre Baxter

Ms Bairbre Baxter is a software developer with a particular interest in usability engineering and the development of web based systems. She has worked for several Irish technology organisations in a systems analyst, software development and support roles. Amongst her qualifications, she holds a Graduate Diploma in Information Technology and an M.Sc. in Information Technology, both from Dublin City University.

CFFES-based Design for Evolving Systems

Urjaswala Vora

Abstract

Architectural Design is one of the most important phases of software engineering process as the quality of the product is decided by the quality of the design process followed. We are focusing on the improvement of architectural design process for the continuously evolving systems. Existing architectural design methodologies need the software systems to be changed even for the evolutions which do not invalidate the existing business logic. That is, the evolution applies to a particular set of entities and the existing architecture addresses other set of entities. Here we are focusing on the evolutions of type MCR, Multiple Concurrent Rules for the business processes. The change in existing system leads to introduction of new errors and raises the risk of regression failure. For the complex systems the coupling among the components is high, hence it becomes impossible to evolve business rules without restructuring large parts of the system. The main concern here is for the software systems for which business processes evolve at high frequency but the systems cannot be taken easily offline due to high costs of their downtime. To address the issue of MCR in such systems we had proposed the framework, Control Flow Framework for Evolving Systems (CFFES). In this paper we discuss the design methodology for systems to be designed based on CFFES so that the software system can lead to seamless evolutions of type MCR.

Keywords

Software Architectures, Framework, Software Evolution.

1 Introduction

Software evolution takes different forms, ranging from fixing errors to implementing functional enhancements, migrations of programs to other platforms, restructuring for ease of maintenance and re-engineering. Evolution of business environments and technological changes affect software requirements and trigger the need for software evolution. However, the speed of software evolution appears to be much slower than the speed of changes in business and technology. After a continuous activity of code modification even the nontrivial combination of small changes affect other parts of program unexpectedly. Those likely ripple-effects of software changes may introduce the potential defects in the updated version. When regression testing fails, it is always difficult for programmers to locate the culprit code by searching through the source. Increasing costs of adapting software to ever changing business needs and technology, makes companies consider software re-engineering as a cure to maintenance problems. Though successful re-engineering projects have been reported, but still the risks and high failure rate of reengineering projects dictate that they do not provide an ultimate solution to software evolution problems.

The existing design methodologies evolved mainly to address these issues arising due to software evolution. The principle of encapsulation in OO design overcame the flaws present in structured methodology due to separate data and process components and their interdependencies. However OO design was proved to be inadequate in addressing design issues due to concerns which are crosscutting across the objects or components or packages. These crosscutting concerns were main obstacles in the evolution of OO systems as code gets tangled across the components or objects.

Aspect-oriented software development was proposed as a solution to the crosscutting problem. Aspect-oriented programming languages are used to modularize the crosscutting structure of concerns such as exception handling, synchronization, performance optimizations, and resource sharing that are cross-cut the source code in existing programming techniques. But Aspect Oriented design also had a few shortcomings such as the components with multiple concerns get decomposed while taking care of each crosscutting concern. This has the negative effect for other concerns requiring the original method. Hence for evolution of type MCR, that is overlapping business rules, the concept of aspect orientation can prove to be more cumbersome than easier to evolve.

In the rule-based systems, externalization of business logic in form of rules was implemented. But the need to embed calls to the rule manager meant that rules are not completely externalised, and much code was required to be examined and modified when business rules changed [11, 12]. Such engines were difficult to be grafted onto an existing software system (especially a database-oriented application).

With emergence of component-based software development (CBSD) technology we can build software systems out of reusable and autonomous application components with well-defined functionality and interfaces. Software evolution can then become an easier task, as we can evolve software systems by modifying or replacing application components. But the criticality of the scenario lied in definition of an autonomous component in a software system which was unanswered by the CBSD technology, since the answer depended primarily on business considerations.

The promise of model-driven engineering (MDE) is that the development and maintenance effort can be reduced by working at the model instead of the code level. The problem with model-driven engineering is that it can lead to a lock-in in the abstractions and generator technology adopted at project initiation. What complicates model-driven engineering is that it requires multiple dimensions of evolution [4]. While MDE has been optimized for regular evolution, presently little or no support exists for metamodel, platform and abstraction evolution.

In Vora and Sarda [22], we proposed a Framework, CFFES (Control Flow Framework for Evolving Systems) that deals with evolution of business rules with overlapping temporal validities. If the existing business rules are valid then the risk of changing the existing operational system should be avoided. Hence the impact of change to the already operational system can practically non-exist and the maintenance costs due to evolution in business rules can reduce by considerable extent. The modularization framework works at the abstraction level of functional components hence can be

applied to systems irrespective of the design methodology followed, e.g. object-oriented or structured. The other support components of the framework are discussed in details in section 2. CFFES components implicitely provide precise results for the supporting tasks of software evolution, which are change impact analysis and evolution prediction.

In this paper section 2 briefs about the framework and its components. Section 3 discusses the design methodology as well as evolution management phases for CFFES-based systems. In section 4 we summarize the phases in the design methodology and evolution management discussed in the paper.

2 Externalisation of Control Flow

We proposed the Architectural framework in [20] and further detailed the same to arrive at following components as shown in figure 1:

(1) Temporal Meta-Data **(2)** Process Controller **(3)** Control Flow Rule Base **(4)** Archiving Engine **(5)** Application Components **(6)** Archived Component Database **(7)** Evolution Manager.

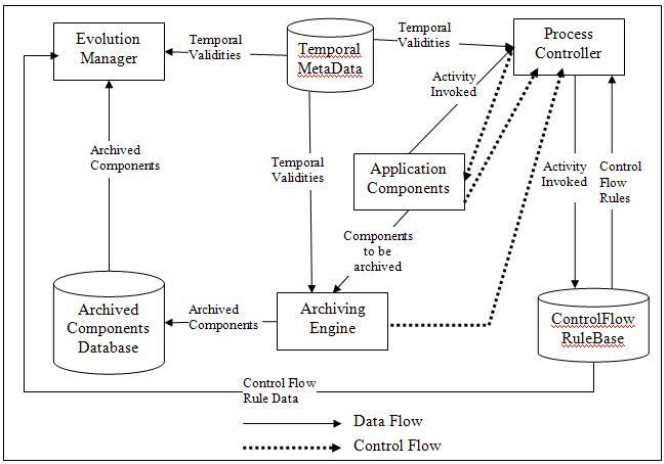


Figure. 1 CFFES

In CFFES, Application Components, Temporal Meta-Data, Control Flow Rule Base (CFRB) and Archived Components Database are application specific components and other components (Process Controller, Evolution Manager and Archiving Engine) are generic components, which are common for any system design. The system design is to be based on the CFFES, Architectural Framework, at the ab initio stage of development and then to manage the evolution effectively. Temporal Meta-Data will contain data about Temporal Validity of data and process components as well as of control flow rules form the CFRB. Application Components include versions of process as well as data components which are designed and developed to provide the application's business processes or activities. At the ab initio stage there will be single version of all the process and data components implementing the business tasks which constitute to the different activities. Activity is to be defined here as the use case realized or the interfacing point (handle) for a user of the system. Activity is composition of business tasks and the control flow rules among these tasks. Each task will be delivered by each version of process component designed to deliver the task, depending upon the control flow rules which in turn are selected depending on the input parameters and/or their values. The Control Flow Rule Base will record the control flow rules to inflict the control flows among these versions of the tasks, for all the activities. The control flow rule will decide the component version for the particular task to be invoked depending on

1. the activity invoked by the user and/or

2. previous component versions for the tasks executed which are part of the activity and/or
3. definition of Input parameters received and/or
4. values of Input Parameters received.

For an existing business task there will be one or more valid business rules defined. The process components which are part of Application Components will implement one or more business rules. Process Component can be defined as implementation of subset of business rules which are applicable for set of particular input parameters to give set of expected output parameters. Data Components constitute the data repository of the application and will be created used and updated by any of the process components. The Process Controller will be the interface between Application Components and the other framework components of CFFES. With the specifications of temporal validities of the versions of process as well as data components in Temporal MetaData, the archiving rules are to be specified for the Archiving Engine. When the particular business rule becomes temporally invalid, the corresponding component versions are archived by the Engine from the Application Components. The control flow rules connecting the component versions being archived also need to be archived from the Control Flow Rule Base, so that the execution environment does not have the load of temporally invalid component versions and the overhead of searching through the temporally invalid rules.

When evolution in the business logic takes place and new business rule is defined for the existing business task, which already has one or more valid business rules implemented. To implement the new business logic, a version of the component implementing MCR (Multiple Concurrent Rules) for the corresponding business task is to be developed implementing the new business rule on the set of required input parameters giving expected result.

As the temporal validities of the versions of components will be different, i.e. usage period of existing business rule and that of the new business rule which came into existence after evolution of the business logic for the activity, are distinct. This information is captured in Temporal Meta-Data Component of the Framework. When evolution takes place, temporal validities of older versions of components, participating in the task, are modified and temporal validities of the new versions of components are added to Temporal MetaData. The addition of new versions of components to the Application Components results into the addition of new control flow rules defining the control-flow among the new versions of components and existing components to the Control Flow Rule Base.

The Framework

1. handles the software evolution effectively without disturbing the existing application architecture.
2. accommodates the concurrent existence of business rules and/or data model for a particular business activity provided by the application software with distinct temporal specifications.
3. maintains temporal specifications of each and every change in the business rule as well as data model and is capable of producing results for different types of temporal queries which can give answers like “what has changed and when”. These types of queries and their results can help in analyzing the evolution of business processes of the organization.
4. can support the statistical analysis of the business processes of the organization, by generating the queries like “what if the change would had varied in …way”.
5. manages the concurrency in the business rules by using the versioning model and applying control flow through Control Flow Rule Base. Temporal validities of the concurrent rules which are overlapping but not necessary are same, are also tapped in the Framework. As the historical data is archived, the analysis of the business processes of the organization can be achieved.
6. supports change impact analysis to arrive at complete and precise change impact set of modules.
7. supports evolution prediction by providing the change history data in the system architecture to precision.

3 Design of Software Systems with CFFES-based Architectures

3.1 Application Components Design

For application components design the requirement is there for aspects to be specified at architectural design level [21]. The aspect specification along with the process components not only modularizes crosscutting concerns but also defines the control flow rules in modular way. As CFFES-based architecture design is based on the concept of externalization of control flow, the definition of aspects results in cleaner control flow which enhances the performance of the application.

3.1.1 Aspects in Architectural Design

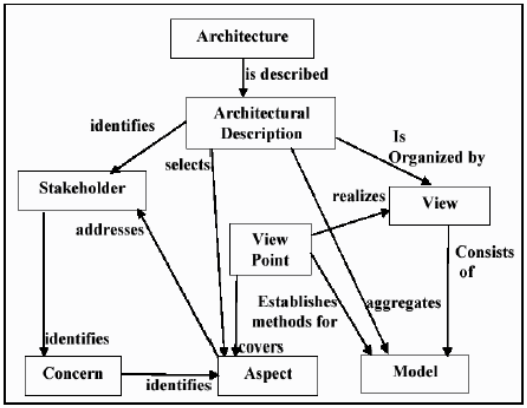


Figure. 2 Conceptual Model for Architectural Description with Aspects

The advantages of applying the idea of aspects to the architectural design descriptions are recognized by Noda and Kishi [14]. As stated by Groher and Baumgarth [5], Aspect Oriented Design is still lacking standardized concepts at the design phase that would foster the specifications of crosscutting concerns at the high level architecture and low level design. The formalization of aspects in the architectural description will fill that gap. For that the conceptual Model for Architectural Description itself needs to accommodate aspects. In figure 2 we introduced aspects as an added layer of abstraction between concerns and viewpoints.

3.1.2 The CFFES-aware ADL

The architecture is to be described using CFFES-aware ADL as well as ADL which treats components and aspects as first class entities alongwith the control flow among them. We have defined the architecture description language (ADL) which is CFFES-aware by -

1. classifying process components and aspects as Activities or Tasks. The definition of an activity and a task is given in Section 2,
2. capturing the version data along with the components/
3. capturing the signature (input and output parameters of the process components),
4. capturing the control flow among the components and aspects.

3.2 Extraction of MetaData for CFFES Repository

After specifying application architecture description using CFFES-aware ADL, we need to extract the metadata about the application architecture to populate the CFFES-metadata repository, i.e. Control Flow Rule Base and Temporal MetaData. The phases required to complete this task are :-

1. Specification of Components and Aspects as Activities or Tasks and capture the Control Flow among them.
2. Architectural Description using CFFES-aware ADL.
3. Conversion of CFFES-aware ADL specification to populate Control Flow Rules in Control Flow Rule Base (CFRB).
4. Specification of Temporal Validities of Process and Data Components to be added into Temporal MetaData component of CFFES. The initialization of the temporal validities of all the components will have the start time to 'forever' validity till the evolution takes place.

3.3 Integration of CFFES-based Generic Components with Application Specific Components

The application architecture consists of Process and Data Components. The detailed design of the same is to be completed and is to be implemented to construct the Application Components (A) part of the CFFES-based system. The steps to be completed in this phase are :-

1. Implementation of Software Architecture to get Application Components.
2. Integration of Application Components with Process Controller.

4 Conclusion

The quality of software process detects the quality of the product being engineered. The quality of the process in turn depends on the quality of each phase of the engineering process. We are focusing on the architectural design process improvement, especially for the software systems which consist of continuously evolving business processes and which can not be put to risk due to high cost of their downtime. When we talk about overlapping evolutions in the business processes, the need for changing existing operational architecture is not there so to disturb the same and add to the risk of ripple effects of the change, is not required. It is estimated 80% of lifetime expenditure on a system is spent on maintenance and evolution. Hence we proposed the framework CFFES for the design of such systems, so that it can lead to seamless evolutions. In this paper we detail out the design phase for evolving systems to be based on CFFES. The framework also provides implicit support for the change impact analysis which is another important phase in life cycle of evolving systems. The precision in results is due to the externalisation of control flow in the Control Flow Rule Base and temporal validities of the versions of process and data components in Temporal Metadata. The evolution prediction mechanism is effectively applied due to metadata of the control flow paths among the process components is maintained in a rule-base model and the temporal metadata of versions of software components in the data model.

We have implemented the CFFES generic components and currently working on case studies from different domains and engineered using different paradigms to be based on CFFES. Our future work involves the change impact analysis and evolution prediction techniques to be applied to CFFES-based software systems and study the results in comparison with the results obtained when the same techniques are applied to the systems following other architectural design paradigms.

5 Literature

1. S. A. Bohner and R. Arnold. Software Change Impact Analysis. IEEE Computer Society Press, USA, 1996.
2. S. A. Bohner. Impact Analysis in the Software Change Process : A Year 2002 Perspective. Proc. of the Int'l Conf. on Software Maintenance, Nov. 1996, pp.42 – 51.
3. Briand, L.C., Wust J. and Lounis, H. Using coupling measurement for impact analysis in object-oriented systems. Proc. of the IEEE International Conf. on Software Maintenance (ICSM), Sept. 1999, pp.475 – 482.
4. van Deursen, E. Visser and J. Warmer. Model-Driven Software Evolution: A Research Agenda. Proc. 1st Int'l Workshop on Model-Driven Software Evolution (MoDSE) (2007), pp. 41-49. University of Nantes.
5. Iris Groher and Thomas Baumgarth. Aspect-Orientation from Design to Code. Proc. of the Workshop on Early Aspects, AOSD Conference 2004.
6. H. Kagdi and J. Maletic. Combining Single-Version and Evolutionary Dependencies for Software-Change Prediction. Fourth Int'l Workshop on Mining Software Repositories, ICSE Workshops MSR 2007. PP 17-17.
7. D. Kung, J. Gao , P. Hsia , F. Wen , Y. Toyoshima and C.Chen. Change Impact Identification in Object Oriented Software Maintenance. Proc. of the Int'l Conf. on Software Maintenance, p.202-211, Sept. 1994.
8. J. Law and G. Rothermel. Whole Program Path-Based Dynamic Impact Analysis. Proc. of the International Conf. on Software Engineering, pp.308-318, 2003.
9. Michelle Lee, A. Jefferson Offutt and Roger T. Alexander. Algorithmic Analysis of the Impacts of Changes to Object-Oriented Software. IEEE, pp. 61-70, 2000.
10. L. Li and A. Offutt. Algorithmic analysis of the impact of changes to object-oriented software. Proc. of the IEEE Int'l Conf. on Software Maintenance, USA, pp 171-184, 1996.
11. Liwen Lin, Suzanne M. Embury and Brian Warboys, Business Rule Evolution and Measures of Business Rule Evolution. IWPSE 2003: 121-125.
12. Liwen Lin, Suzanne M. Embury, Brian Warboys. Facilitating the Implementation and Evolution of Business Rules". ICSM 2005, Proc. of 21st IEEE International Conf. on Software Maintenance: 609-612.
13. Navasa, A., Prez, M.A., Murillo, J.M., and Hernndez, J. Aspect Oriented Software Architecture: a Structural Perspective. AOSD 2002.
14. N. Noda and T. Kishi. On Aspect-Oriented Design: An Approach to Designing Quality Attributes. Proc. of 6th Asia Pacific Software Engineering Conf., pp. 230-237, 1999.
15. A. Orso, T. Apiwattanapong, and M. Harrold. Leveraging Field Data For Impact Analysis And Regression Testing. Proc. of European Software Engineering Conf. And ACM Symposium On the Foundations Of Software Engineering (ESEC/FSE'03), Finland, September 2003.
16. J. Andrés Díaz Pace and Marcelo R. Campo , Analyzing the role of aspects in software design. Communications of the ACM, Volume 44, Issue 10 (October 2001), Pages: 66-73.
17. M. Pinto, L. Fuentes and J. Troya. DAOP-ADL: An Architecture Description Language for Dynamic Component and Aspect-Based Development. 2003. Proc. of the 2nd Int'l Conf. on Generative Programming and Component Engineering, Germany. PP. 118 - 137.
18. H. Sneed. The Drawbacks of model-driven Software Evolution. Proc. 1st Int'l Workshop on Model-Driven Software Evolution (MoDSE) (2007), pp. 41-49. University of Nantes.
19. Tip, F. A survey of program slicing techniques. J. of Programming Languages 3, 3 (1995), 121-189.
20. U. Vora. Architectural Design Methodologies for Complex Evolving Systems. 12th IEEE Int'l Conf. on Engineering of Complex Computer Systems (ICECCS), New Zealand July, 2007.
21. U. Vora. Aspects in Architectural Description of Evolving Systems. First Workshop on Aspects in Architectural Description, held at Sixth Int'l Conf. on Aspect-Oriented Software Development, British Columbia, March 2007.
22. U. Vora and N. L. Sarda. Architectural Design Issues for Evolving Systems. WSEAS (World Scientific and Engineering Academy and Society) Transactions on Systems, Issue 4, Volume 5, April 2006, ISSN 1109-2777.

6 Author CV

Urjaswala Vora

Urjaswala Vora received her Engineering Degree in Computer Science from the University of Pune, India (1994). She is working in Centre for Development of Advanced Computing (CDAC), Mumbai as Senior Staff Scientist since June 1997. Her research interests include Software Design Methodologies, Software Architecture, Software Evolution and Design Modeling. She is a research scholar with Computer Science and Engineering Department of Indian Institute of Technology (IIT), Mumbai, India. She is on the International Domain Experts Board of the International Journal of Software Architecture as well as on the reviewer list for IEEE 1471 - Recommended Practice on Software System Architecture Description.

Success Factors for Globally Distributed Projects

Siegfried Zopf
Software Engineering & Qualitätsmanagement Zopf
Siegfried.Zopf@sqz.at

Abstract

The report presents the essence of experience from hundreds of development projects distributed over locations in eight countries with different languages and culture. One success factor is professional project management based on a well defined development process. The report highlights the topics that have special importance in distributed project. Practical tips will be given for how to deal with soft facts as esteem, team building in global projects and traps in communication in foreign languages and how to avoid them.

Keywords

Outsourcing, global development, distributed development, CMMI, development process, quality assurance, human factors

1 Introduction

1.1 Scope of outsourcing

According to a survey of Cutters Consortium more than 80% of the Fortune 500 companies outsource software development. 86% of them go offshore. [Herr03]

In most cases cost cutting is the reason for outsourcing.

- Cost (44%)
- Capacity (20%)
- Know how (13%)
- Time to market (11%)

As there are different reasons for outsourcing or globally distributed projects there are different ways to organize those projects. I will distinguish between:

- **Distributed projects** within one company, where the responsibility for the project is in the company and the execution of development takes place in some locations may be with local responsibility for subprojects.
- **Outsourcing to another company** may be offshore or - as it is often the case in Europe - to other countries with lower hourly rates on the same continent. Additionally to the things that concern distributed projects come activities concerning selection of a supplier / partner, contractual topics and customer – supplier relationship.

In both cases there are different situations depending on the extent of outsourcing from “headquarters” to a “remote location” be it in the own company or third party. [Cam05]

Minimal responsibility outsourcing

The headquarters decides about project goals, elaborates the requirements, designs the product and the system architecture and is in charge of all management aspects like project management, quality assurance management and configuration management.

The remote location makes the detailed design (if at all), the coding and component test.

System integration, system test and acceptance is done back home in the headquarters.

With minimal responsibility outsourcing the domain knowledge as well as the software engineering knowledge persists within the headquarters. As coding is only a small part of a development project the advantages are relatively small but also the risk is small and in case of problems it is not so difficult to in-source. Even with minimal responsibility outsourcing the experience with implementation (all those information that is not documented about tools, platforms etc.) does not come back to the designers and the design becomes less realistic over time.

Large scale outsourcing

The headquarters decides about project goals, elaborates the requirements and mostly the design of the product. Also overall project management and quality assurance management rests in the headquarters. The whole technical jobs like system architecture, detailed design, coding, integration and testing are executed at the remote location which is responsible for the development including subproject management. At the end of the project headquarters only execute the acceptance test of the system tested product.

This model contains the big danger for the headquarters of loosing the qualification for the technical

job which may lead to dependency on the supplier. Know how is transferred to the supplier. This model should not be applied for strategically important software because the strategic advantage can get lost from one day to the other when the supplier changes his partner.

The risk of loosing technological know how can be reduced when one or more subprojects are still done at the headquarters. In case of skipping the contacts with a supplier there is still a competence for development and maintenance at the headquarters.

I will not discuss the model where you send a requirement spec to a supplier and wait for the product to be delivered in time.

1.2 Role of a Beach head

Even in large scale outsourcing the headquarters have to have beach head functionality for controlling the remote location during the development phases. When it is a large scale outsourcing project some make the error to think: „we have sent all the requirements to our partner and do not need time and capacity for the project any more".

2 Development methods matter

Using a well defined development process always matters, even if you work in one location. But it becomes more important in distributed development, independent of the type of cooperation. [Kain98], [Höhn08]

2.1 Definition of the Product

In large scale outsourcing projects you need a well specified requirements document because the customer or user usually is not on site at the remote location. As we will discuss later there are many reasons why remote developers often do not clarify ambiguous specs or open topics in specs but try to implement something they think is a good solution. In minimal responsibility outsourcing projects the same is true for architectural specifications. Long time ago we saw that the waterfall life cycle model in its purest form did not work. Only documents were handed over from the experts of one phase to the experts of the next phase without additional communication e.g. requirements engineers hand over the requirement spec to the architects. Why should it then work when the developers are on different locations?

Even with the best specifications there will rest ambiguities and open questions. The worst case is that the developers make assumptions: The good case is that they ask questions. But for this they have to have someone who is able to answer. Therefore a "beach head" is necessary.

2.2 Project Management

A clear documented definition of the responsibilities in projects is indispensable. Is everything controlled by the beach head or do you install subprojects. It is nothing new compared with a project with subprojects that you do in one building. But it is more important because you detect problems later, when they have become more expensive. And it concerns not only project planning and control but also technical roles, quality assurance management, configuration management, test etc. [Früh01], [PMI05]

2.3 Risk Management

Especially with new cooperation partners not everything will run as you think. Therefore risk management is more important in distributed projects then in the projects you are used to do.

Frequent sources of risks are: communication, langue difficulties, cultural difference, tacit assumptions, knowledge of application domain, understanding the common process (if the remote developers have to use the process of the beach head). This comes in addition to the usual risks. [DeMa03]

2.4 Quality Assurance

Quality assurance is another one of the topics that would be important in projects but are neglected very often. In distributed projects quality assurance and the role of a quality assurance manager (QAM) becomes crucial. [Wall01] Depending on the type of cooperation you have either only one Quality Assurance Manager (QAM) in the beach head or you have an additional local QAM in each remote location. In the first case it will be very difficult for the QAM to get reach insights how the developers do their job, how they stick to the processes – in reality, not only the formalisms. A local QAM has the chance to see early indicators of problems. It is also easier to organize quality assurance measures like reviews and observe how they are done.

When you have one QAM for all locations he/she can call in the documentation of quality assurance measures and get an impression on how the project runs. Therefore he/she has to study the documents and not only collect them. Review Report: Are peer reviews done professionally? Was the review object o.k.? Test plan: How are the tests planned? Test report: Have the planed test cases been executed and how many defects were detected etc.

2.5 Configuration Management

I was very surprised when I attended a presentation at an international conference in Düsseldorf 2006 about optimisation of distributed projects, an experience report. The big thing they did was introducing Configuration Management (CM) in the project. And it really helped and solved a lot of problems they had before. I was surprised because I thought CM is standard for 20 years but it showed again that the engineering discipline is not yet basic knowledge in software development organisations. [Berl92] What is important for us is that a distributed project without CM runs into troubles. You do not need risk management, you have a problem.

3 Is CMMI Maturity Level 5 the Silver Bullet?

The Capability Maturity Model Integration (CMMI) is a reference model for the maturity of software development organisations with five maturity levels. It was elaborated at the Software Engineering Institute (SEI) at the Carnegie Mellon University [SEI06], [Hump89]. CMMI is a quai standard all over the world. In appraisals the capability of a supplier can be assessed.

When the development process is so important many contractors have the idea: let's look at the CMMI maturity level of the supplier and select the one with the best ML.

The Idea is good but there are some constraints. Does the partner really work on the promoted maturity level (ML)? Has the beach head the maturity to cooperate with a high level organisation?

3.1 Maturity Level of a development organization

Be careful when you get the informal information “We are on ML5”. What part of the organization was appraised? The department that you work with or another department that may be only 10% or less of the organization and therefore the ML is not typical for all projects. Another question is: who did the appraisal? An appraiser accredited by SEI or someone else may be even the own organization in a self appraisal. Therefore it makes sense to look at the certificate and collect additional information about the partner. Best is to contact clients of the supplier.

When you have an understanding of CMMI you will see in the first days of the project what level the partner has.

3.2 Maturity Level of the beach head

Employing an organization with ML5 only makes sense when the beach head itself has at least a good understanding of ML3. Otherwise the partner will not be able to proceed as required by CMMI ML5. In project cooperation the leading part has to understand and act on ML3. It does not work when the beach head says: ”We do not have the money for CMMI here in Europe. We do our part as we have done all the time but the others shall develop the outsourced part on ML5. They have cheap developers and have the time and capacity to do everything required by CMMI.”

4 People are People in all Corners of the World

Software development is a people business not a machine business. People are the most important “resources” in a development project. We know that the success of development projects depends more on motivation, team work and communication than on hardware, programming languages and development processes. The development department Siemens PSE, where I spent my professional carrier until recently, has 20 development locations in 8 countries of the world. Globally distributed projects are the rule, not the exemption. That is why big effort was spent to find out what makes distributed projects successful. [Acke00]

Motivation, team work and communication are impaired strongest in distributed projects. Hardware and software tools are nearly not influenced.

To be motivated you have to know the goals, have to be included in decision making and have to get recognition by the team and by the management. Motivation and team building may arise sometimes even without much systematic effort and investment of money when the team is in one place. But in distributed teams it never arises without systematic effort and it costs money. A kick off meeting produces additional travelling costs. When you think you can safe this money try it. But measure the failure cost of the project. Analyse the date and the next time you will invest in team building.

Cultural aspects play an important role. Therefore inform yourself about the differences. E.g. in some cultures a person usually will not say “no” but will use other ways to express his disagreement: e.g. “there is another possibility to do this”. Therefore “he did not say no” is not an agreement. Listen to what he says and think about it.

Esteem is important for people in all corners of the world. To esteem your partner is the precondition for successful collaboration. It has the same importance in a local project. But to esteem someone you know, who has the same lifestyle, the same education, loves music and actors you also know comes sometimes automatically and is easier than to esteem someone who is different in many aspects. In this case it does not come automatically and you have to strive for understanding. Care to get information about the interests and achievement of someone and also the circumstances of his life. Be careful not to awake bad feelings because of the different levels of standard of living. But the esteem has to be real. Nobody can play it like an actor. Body language, accent and the like will betray you. But esteem is the basis for successful communication.

5 Communication

In the flow of information through the project as well as in team building language is in the centre. One or sometimes both partners have to communicate in a foreign language. Usually the ability to express thoughts and feelings in a foreign language is lower than in the native language. Also the understanding of thoughts and other messages is lower. Therefore in a globally distributed project attention has to be spent on communication. It is an important task of the management to look if the formal communication runs well. Every team member has to be aware of this challenge and has to be alert in everyday communication written or verbal.

In many cases written communication is better than verbal. The partner can analyse the text with a dictionary what he cannot do after a conversation. But there will be conversations and the question is how to check if the message did arrive. “Did you understand?” is a stupid question. Nearly everybody will say yes. Better is mirroring. “What did you understand?” and than the partner will explain with his own words what he understood. Then you know if he understood or not. It sounds a little bit like Imago Therapy but it works. I do not object to conversations and it is good practice to write down the result of the talk in file note. In cross cultural projects it is useful to follow this practice, write a file note and additionally send your not to your partner.

To be sure observe what the partner does shortly afterwards. Do not wait 3 month if he will bring back the right thing but check after one week if he is on the right way.

6 Summary

In a nut shell the success factors for globally distributed projects are not rocket science:

The most important success factor is professional project management based on a well defined development process.

The architecture of the system and distribution of tasks to different locations should be mapping as much as reasonable possible.

Optimizations have to treat always the global project not the local parts. Watch for local optimizations at the cost of other sites and turn it around.

Plan higher effort (beach head, laborious communication…) than you would for a local project. Higher effort does not mean higher costs. But planning a distributed project with the same effort that was estimated for execution in one location will generate problems and higher costs.

Be always aware of cultural differences and put more emphasis on systematic measures and activities that cultivate human factors and never forget esteem.

Literature

[Acke00] *Ackermann Gerhard / Lutz Benedikt, Cross-Regional Cooperation in PSE Projects, Best Practice Report, Siemens internal paper, 2000*

[Berl92] *Berlack H.R., Software Configuration Management, John Wiley &Sons,1992*

[Carm05] *Carmel Erran, Offshoring Information Technology, Cambridge University Press, 2005*

[DeMa03] *DeMarco Tom / Lister Timothy, Waltzing with Bears, Dorset House, 2003*

[Früh01] *Frühauf Karol / Ludewig Jochen / Sandmayr Helmut, Software-Projektmanagement und -Qualitätssicherung, vdf Hochschulverlag, 2001*

[Herr03] *Herron David, The Impact of CMM on Outsourced Software Development, Cutter Consortium, Executive Report Vol.4, No.5, 2003*

[Höhn08] *Höhn Reinhard / Höppner Stephan, Das V-Modell XT, Springer, 2008*

[Hump89] *Humphrey Watts S., Managing the Software Process, Addison_Wesley,1989*

[IEEE04] *SWEBOK Guide to the Software Engineering Body of Knowledge, IEEE, 2004*

[Kain98] *KaindI Hermann / Lutz Benedikt / Tippold Peter, Methodik der Softwareentwicklung, Vieweg, 1998*

[PMI05] *Project Management Institute, A Guide to the Project Management body of Knowledge, PMI Books, 2005*

[SEI06] *Software Engineering Institue, CMMI for Development, Version 1.2, CMU/SEI 2006-TR-008 ESC-TR-2006-008*

[Wall01] *Wallmüller Ernest, Software Qualitätsmangement in der Praxis, Hanser, 2001*

Author CVs

Dipl. Ing. Siegfried Zopf

Mr. Siegfried Zopf is self employed consultant for software engineering and quality management. He studied technical physics at the Vienna University of Technology in and started his professional carrier as software developer with Siemens Program and System Engineering (PSE) in 1977. In 1984 he changed to the central quality management department of PSE where he participated in the elaboration of the system development method SEM. He was responsible for the implementation of SEM and the advancement of software engineering and quality management in PSE (ISO 9001 certification, CMMI and EFQM Assessments) at 20 locations in 8 countries.
Besides he is lector at the Vienna University of Technology and at universities of applied science. Since 2005 he is president of the Austrian Association for Software Quality

Managing Globally Distributed Software Development in Schlumberger

Vibeke Dalberg, Lars Erik Mangset

Det Norske Veritas, DNV Research and Innovation

vibeke.dalberg@dnv.com, lars.erik.mangset@dnv.com

Richard Davies, Vincent Dury

Schlumberger

rdavies@slb.com, vdury@slb.com

Abstract

This paper introduces two largely distributed software development projects in Schlumberger, documents key learning points and summarizes best practices for working on distributed projects in Schlumberger.

Keywords

Distributed software development, global software work, global software engineering, process improvement, best practices.

1 Introduction

As an international company, Schlumberger often establishes projects across multiple sites within the organization. This is due to, for example, more effective market positioning of products and services, cost reduction, closeness to business stakeholders and access to domain expertise.

Global business demand local adaptation of product and services, in order to stay competitive in the marketplace. In technology intensive organisations such as Schlumberger, this involves running global software development teams, which run parallel with local product development.

Such global software work implies overcoming certain barriers that hampers effective and innovative projects. This paper present some challenges experienced in large and complex Schlumberger projects, and summarize some of the identified best practices.

1.1 Background

This study presents findings based on seven interviews with project members of two large distributed software development projects in Schlumberger, including the experiences of the Schlumberger co-writers and reviewers. The interviews were semi-structured, and were conducted during the fall of

2007 by researchers from DNV Research and Innovation.

This paper focus on internal distribution, while the paper Jensen, Menon, Mangset, Dalberg (2007) [1] focuses on Schlumberger offshore outsourcing in more detail.

This paper is not to be considered as an exhaustive solution to all the various challenges encountered when working on distributed software development projects. Its goal is rather to document and provide Schlumberger best practices from which to learn - and to possibly influence future decision making processes.

1.2 Examples of distributed software development projects in Schlumberger

We interviewed project members in two large distributed software development projects in Schlumberger; "Alfa" and "Beta". In this chapter we give a quick introduction to the relevant aspects of these projects.

Example 1 – Alfa:

The Alfa project was initiated in 2003, with the main objectives of improving peoples' efficiency, service quality and consistency. It delivers field personnel engineering workflows defined by domain experts and business teams. Alfa replaces legacy engineering applications and locally developed Excel spreadsheets created by field engineers.

The project organization is truly distributed across various Schlumberger centres (Sugar Land, Paris, Novosibirsk, Beijing), as well as involving several outsourcing partners (College Station, Novosibirsk, Bangalore) both locally and offshore. See figure.

The Alfa project is regarded as one Program. It is a single Alfa application installed by the end user, and a single Alfa Program organization. However, each workflow available from Alfa is developed as a different project and is assigned to one team in one centre. All projects/workflows must use the same infrastructure and share the same components. Extensive collaboration between the centres is required. Most GUI development is done in Sugar Land while other development requiring specific domain knowledge may be implemented in Novosibirsk or Paris.

Example 2 – Beta:

The Beta project was initiated in order to replace 120 wireline applications and 40 drilling and measurements applications.

At its height, the project had about 250 project members. The project is highly geographically distributed across several Schlumberger centres (Sugar Land, Austin, Princeton, Paris, Beijing, and Tokyo) and related outsourcing partners (Grenoble, Tokyo, and Bangalore). Austin was in 2005 replaced by Beijing. See figure.

Beijing is responsible for the system development, while Sugar Land, Princeton, Paris, and Tokyo are responsible for application development. Program management was located in Austin, and later moved to Beijing.

In the remainder of this paper, we will refer to Alfa and Beta as programs, each comprising several projects.

2 Schlumberger experiences

Working in distributed rather than co-located projects introduces a range of challenges. Geographical, temporal, cultural and environmental factors all result in collaboration, communication, and coordination issues becoming much more difficult to handle.

2.1 Coordination

The management of the program is regarded as important both for the internal program decision process and collaboration, as well as for the external visibility and presence. This top management is centrally funded, while the various teams are funded by each centre.

2.1.1 Roles

All software development projects in Schlumberger have similar roles; however some are more influenced by the distribution than others. Important roles in a distributed project that are affected by the distribution dimension are: Program lead, Product champion, Chief architect, Project manager, Product manager, Technical lead, and Developer. A clear definition of these roles ensures transparency for all team members supporting effective communication and redistribution of responsibilities to other team members.

Program Lead

For distributed development teams in Schlumberger it is important to have a horizontal management layer that will keep focus on the overall program vision across all the vertical project objectives.

The role of the program lead is to overlook the entire program, both internally and towards stakeholders. It is an important role in terms of creating visibility, coordination and communication. Schlumberger management changes quite often so a program lead can maintain consistent communication regardless of this regular change. Such a role is highly important in large, distributed projects. A program lead is a role that can be filled by a program manager or distributed among several people. The latter will imply increased complexity related to coordination, collaboration and communication in carrying out the program lead role in a distributed project. The program lead's main responsibilities would be to facilitate the creation and maintenance of the overall vision and roadmap, and to continuously communicate it to the distributed team and stakeholders. Other responsibilities of the program lead are to coordinate all the projects and products in the program, to arrange common meetings and workshops, as well as to deal with budgeting, planning, and general direction decisions.

Both projects examined were currently lacking a program manager carrying out the program lead role, and this is seen as a major and critical weakness for one of the projects. However it might seem easier to cope without top management when the project is well established, the structure is in place, and everybody knows each other, than it is on the other project.

2.1.2 Managing the vision

Given the extra challenges encountered in a distributed software development program (i.e. consisting of several projects), it is essential that all the parties involved move in the same direction. Therefore it is important to have a strong vision of what needs to be achieved, and that everyone understands it and adheres to it, as this eases coordination, collaboration and communication.

In one of the investigated projects, the business teams defined business deliverables and the engineering teams defined the high level architecture and sub systems, and identified technical risks. The combination of these elements enabled the generation of *the Development Roadmap*.

The roadmap and the architecture were used to identify the major components of the system and to define where they could be developed according to the different competencies across the distributed project. The roadmap provides a good coordination tool to identify similar development efforts in dif-

ferent parts of the roadmap. The roadmap is a good tool to manage changes. It helps understand how the changes (business, organization, people, technology, politic, etc.) impact the deliverables and how the project can be reorganized to reach its objectives. It is usually used to prevent some small changes that might be locally beneficial for a project, but that can have catastrophic impact on the overall vision of a globally distributed project. It federates all the distributed development teams to the same overall vision and objectives and not only to their local projects.

2.2 Collaboration and communication

Collaboration and communication structures are highly affected by a distributed setting, and the processes will have to carefully be adapted to this setting. Distribution across geographic borders, time zones, and various cultures will affect the project's ability to collaborate and communicate. One of the management members estimated that communication in a distributed project takes about three times more compared to communication in a co-located work setting.

2.2.1 Defining how to work together

It is essential to define how the different teams will work and communicate together. The two main approaches are:

1. To work as *separate entities* like different contractual organizations.
2. To work as *a single* big distributed team.

Working as separate entities requires that the dependencies between the teams are formalized. Well defined interfaces result in a lesser need for tight coordination between the teams – indeed each team can have its own process and organization. However, using this approach each team focuses on its own objectives, and not on realization of the overall vision. If all teams work together as a single agile team, it may be easier through shared team goals to achieve the long term vision. Continuous common planning and change management enables faster feedback loops, and delivering the greatest value product earliest.

As an illustration, the Alfa project uses this second approach. One main objective was integration between legacy engineering applications. Previous experience with legacy applications showed that distributed development has led to separate applications, while the software architecture was actually ready to support the integration. By having a single team with a common objective to deliver a single application, the architectural integrity of the system was maintained.

2.2.2 Time zone differences

With large time difference within a project e.g. between Sugar Land and Novosibirsk, there may be no overlapping working hours, challenging the coordination across the entire program.

A very large time difference between collaborating centres will also affect the collaboration within the teams i.e. because the communication has to happen through asynchronous tools rather than synchronously. Asynchronous communication such as e-mail typically delays a response for two key reasons. Firstly, time differences often result in responses not arriving until the next day. Secondly, simply by not knowing or seeing your remote colleagues physically can cause a natural down-prioritizing of the e-mail increasing the risk of further delays.

2.2.3 Relationship and trust

Building relationships and trust across borders in a distributed team is very important. This is necessary in order to understand each other and each other's working style and to overcome any cultural differences.

There is a tendency that the person in a team that has visited and knows the other team operates as the main liaison with this other team, as this enables quicker responses. This can however also create bottle necks.

The program manager of one of the programs used to call each project manager individually once a week, in order to create an atmosphere more open than in the regular common meetings. These phone calls were regarded as very important and successful. In one of the projects reported in this whitepaper a map including pictures of each individual on the project team was created. This was distributed to the whole project, was brought to phone meetings etc., and served as an important artefact for people to feel closer to each other.

2.2.4 Meetings

In geographically distributed projects, meeting points become crucial in order to coordinate, communicate and collaborate across geographical borders and time zones. Both regularly co-located meetings (demanding travel), and distributed phone meetings are used extensively.

Examples of typical phone meetings in the two investigated projects are:

- *Stand-up meeting.* Every morning the project that runs agile development has a stand-up meeting between US and France.
- *Technical meeting.* Both projects have a technical meeting every week, but one of them have such a large time difference between the centres, so they have to run it twice. One of the projects did not use to have such a meeting, which resulted in e-mail correspondence, and therefore asynchronous response.
- *Management meeting.* Weekly meetings for all managers involved in the Beta project.
- *Committee meeting.* Every third week the Alfa project managers and technical leads meets.

Regardless what time of the day it is, one of the investigated projects emphasized not exceeding one hour for the duration of a phone meeting. In addition to these meetings physical face-to-face meetings are also held such as:

- *Technical meeting.* The regular face-to-face technical meetings for technical leads and project managers are regarded as important in aligning the Alfa project as one team and for solving issues in high need of face- to-face discussions. Identification of re-useable components is typically done in these meetings.
- *Project manager meeting.* One of the projects gathers the whole management team every third month, while the other meets every sixth month.
- *Product champion meeting.* A product champion meeting is held at the management location every quarter.

2.3 Software development methods

As distributed projects grow, the dependencies between the participating centres and teams increase. Both investigated projects have highly interdependent teams across centres around the globe, and at the same time focus on having common release dates. The dependencies can cause delays due to having to wait for responses from other centres, teams, the management, the champions etc.

Both investigated projects run development iterations on 2, 3 or 8 weeks, and report good experiences. One person recommended shorter iterations in order to improve better feedback loops resulting in faster realization of short term goals. If the time-span is too large, the momentum, focus, and dedication for the iteration goals may decrease. Working distributed challenges the “one-software-method” approach. Experience shows that there are differences in software development cultures, and technical conflicts are harder to manage, requiring more effort.

Many project managers and developers naturally want to reduce dependencies and therefore prefer traditional waterfall process that encourages the rejection of change requests. One of the projects runs an agile software development environment, which emphasize personal communication. Their experience is that not all parties in the project prefer agile methods, and would rather like a more traditional structure.

Component re-use is a focus, but the distributed teams and development makes it more challenging to discover and communicate re-useable components. Some processes are in place to support this in one of the projects, for example through face-to-face meetings.

2.3.1 To distribute or not…

Some software development activities are best done in a co-located manner, while other activities can easily be distributed. Most of them are somewhere in the middle. Some examples of what could be done distributed based on the experiences from the reported Schlumberger projects:

- Everything that is purely technical, for example data migration.
- Testing and bug fixing
- All development activities for separate components where you can define the interface.
- Software re-engineering.

Some examples of what should be considered not to be done distributed:

- Tasks that entail high need for domain knowledge. It is hard to grasp all the details, understand and anticipate the different scenarios that can occur, unless you are co-located.
- The development of the system framework.
- When there are loose boundaries and many interfaces.

2.4 Handling internal and external change factors

Programs continuously face internal and external change factors that must be managed. The nature, number, and impact of such factors typically increases in distributed projects compared to single-centre projects. The distribution of a project is usually decided based on business factors, such as access to domain expertise, marketing presence, development cost, etc. The development team usually has little control over this.

Internal political distribution is a challenge to take into consideration. For example, Alfa teams were each belonging to different sub- organizations developing specific workflows for each of them. The Alfa program therefore had to deal with a lot of different stakeholders, and it was difficult to find agreements on common requirements between parties.

The team should not expect the external and internal factors to be stable during the life of the project. Everything is subject to change; that could for example be promotions, transfers, attrition etc. among project members, management team, or among stakeholders. It could also be organizational changes, such as internal reorganization, acquisitions of new companies, business ventures with other companies etc. Technology can change, and local regulations can change. The project has to be ready to handle these changes, and a distributed organization makes this more challenging.

One of the investigated projects experienced that their Austin centre was moved to Beijing in the middle of the project. Schlumberger was motivated both by having an open presence in China, and to turning the Beijing centre into a software centre of excellence, according to our interviewed respondents. The reported experience of this has been that in general terms it largely affects the progress of the project, for shorter or longer time periods. It also results in a loss of momentum, since only a few of the team members in Austin moved with the project to Beijing. The experience of losing the program manager in the middle of the project, and not being provided a replacement, was experienced as really challenging.

Change can also be a benefit. One of Schlumberger strengths is the diversity of its employees. It is part of the company culture to move employees between centres regularly. This serves to increase diversity, people bring with them the knowledge acquired from the previous centre and to challenge what is in place, improving knowledge transfer and innovation. In Alfa, one of the project managers moved from US to France, consequently bringing with him the Sugar Land centre culture and know-how, as well as knowing the people at both centres. Experience showed that this served to improve the communication between these two sites.

3 Schlumberger best practices

This chapter summarizes the best practices identified for internally distributed software development in Schlumberger.

Collaboration and communication in distributed projects:

- Program manager should call each project manager once a week.
- Ensure that effective communication channels are developed, maintained and pro-actively improved between the distributed teams.
- When receiving an e-mail request, try to prioritize this. At the very least, respond immediately with "I will get back to you shortly". Encourage a very high standard of e-mail etiquette and discipline.
- Distribute pictures of all team members, for example through a picture-map.
- All team members, both management and developers, should meet face-to-face with the people they are supposed to collaborate with.
- Make sure to hold regularly meetings where the whole project can meet. Encourage a very high standard of meeting etiquette and discipline.

Coordination of distributed projects:

- When establishing distributed projects, locate the project management geographically as close to the "centre of mass" of the participating centres, to ensure the most effective communication Possible.
- Have one program manager in the program lead role for large distributed projects to ensure visibility, coordination, communication, and vision.

Software development methods in distributed projects:

- Consider as little interdependence as possible between the centres or teams, for example through modularization of the development.
- Run short iterations in order to keep momentum and focus.
- Run short iterations in order to more easily keep track of the progress and the quality of the work.
- When choosing software development method for a globally distributed team, consider the experience, traditions and preferences of the involved parties towards different methods and working styles. Training may be needed.
- Modularization might open up for multiple software development methods within one project.
- Re-use of components across the program may support a more coherent end result for lower cost, but it requires attention and focus across the distributed program.

Tools to support coordination, collaboration and communication in distributed projects:

- Globally distributed teams need both synchronous and asynchronous tools to support their communication and collaboration.
- Easy upgrades, such as a new microphone or better bandwidth can ease the collaboration experience in teams dependent on electronical collaboration.

4 Summary and Conclusion

Working with distributed teams brings many challenges, mainly concerning practical coordination, collaboration and communication issues. Change management is a key focus area. Such organizations demand a clear vision and development roadmap, and the right people to manage them. A successfully functioning distributed project will benefit from knowledge sharing and the cultural diversity, often resulting in enhanced creativity and innovation. The lessons learned and best practices described in this paper will be promoted through Schlumberger's Software Métier organization to reduce the risk of repeating less successful ventures in the future. We will also be able to present these findings to project sponsors, in order to steer future decisions regarding project organisation.

5 Literature

[1] Jensen, Menon, Mangset, Dalberg. Managing Offshore Outsourcing of Knowledge-intensive Projects - A People Centric Approach. ICGSE 2007, Munich, August 2007.

6 Author CVs

Vibeke Dalberg

Vibeke Dalberg is a Senior Research Scientist at Det Norske Veritas, DNV Research and Innovation. Her main research focus has the latest years been on global software work, work processes, and process modelling. Dalberg has been involved in global software work both from the practical project side, as well as from a researcher perspective for about five years. She has been in the Program Committee of ICGSE 2007 (International Conference on Global Software Engineering), and in the Organization Committee of the workshop "The Challenges of Collaborative Work in Global Software Development" at ECSCW 2007 (European Conference on Computer Supported Cooperative Work). Dalberg has been the project manager of the joint industry project performing the Schlumberger study presented in this paper.

Lars Erik Mangset

Lars Erik Mangset is a researcher at the Det Norske Veritas, DNV Research and Innovation, where he has focused on global software work, work processes and risk management. Mangset has been a project member in the joint industry project performing the Schlumberger study presented in this paper.

Richard Davies

Richard Davies PMP is Software Métier Manager at Schlumberger's Oslo Technology Center, where his focus is on people, process, technology and training. Starting his career in Schlumberger as an operation's geophysicist coordinating land seismic surveys, his domain knowledge and computing science education led to a software engineering position in Schlumberger's Research & Development organization. Working on large, multi-national and multi-discipline teams he gained experience leading projects covering data recording control, Geographical Information Systems and manufacturing test systems distributed across various product centers worldwide. Qualified as a PMI Project Manager Professional, in 2005 Richard then spent 2 years in Texas managing business systems development for Schlumberger Drilling and Measurements. These projects serve over 150 operational bases worldwide, with stakeholders in operations, engineering and manufacturing – and development teams in Texas and India. He enjoys the challenge of developing the engineering, manufacturing and sustaining software organization to meet the company's needs, and setting out strategy for future success!

Vincent Dury

Vincent Dury is Project Architect at Schlumberger Information Solutions in Oslo. He started to work as software contractor in 1988 on short missions for different companies and domains. He joined Schlumberger Well Services in 1992 to develop, design and manage software applications to simulate and optimize Drilling Fluids and Cementing operations. In 2001 he moved to Texas to manage a multi-disciplinary team developing programs to model Acidizing and Fracturing operations. In 2004 he took the position of Program Manager to integrate all these simulators into a single program providing guided work flows to the end users, coordinating the efforts of teams in USA, France, Russia, China and India. In 2007 he moved to Oslo as Ocean Petrel Project Architect where he challenges technical decisions and mitigates risks with developers located in Norway, England and Hungary.

A Structured Approach to Global Software Development

Valentine Casey & Ita Richardson

Abstract

The analysis of the combined results from three independent industry focused case studies, undertaken in the area of distributed software development over a period of eight years, has resulted in the identification of ten key factors. These ten factors have been utilised as the basis for the development of the *GSD Implementation Model*. The objective of the creation and presentation of this model is to provide a practical and systematic approach to address the key activities, infrastructure and support which are required to facilitate effective distributed software development. This approach is inspired by the IDEAL model and divided into five specific phases which are classified as Initiating, Provisioning, Establishing, Managing and Leveraging. The goal of the Initiating phase is to clearly determine why, if and how the distributed development strategy is to be selected and undertaken. The implementation of the Provisioning phase is to ensure that the required infrastructure, processes and support to facilitate successful distributed software development are identified and put in place. The focus of the Establishing phase is to ensure that the development teams are effectively established. The managing phase addresses the day to day requirements of operating efficiently in a distributed environment. The Leveraging phase concentrates on the need to ensure that the structures and procedures are in place so that lessons learned can be documented and leveraged in existing and future projects.

Keywords

Global Software Development, GSD, Virtual Teams, Outsourcing, Offshoring, Infrastructure, Software Process, Risk, Project Management, Culture, Communication, Coordination.

1 Introduction

In today's highly integrated international markets software development is considered a globally sourced commodity [1]. The sustained popularity for the selection of this strategy is ascribed to organisations endeavouring to gain and maintain competitive advantage from the globalization of software development [2]. The potential for achieving this advantage is attributed to the benefits provided by labour arbitrage, which offers the opportunity for reduced development costs [3]. This continues to be facilitated by the availability of well educated and technically competent software engineers in low cost centres in Eastern Europe, Latin America, India and the Far East [4, 5]. It is a commonly held belief that these savings can be coupled with the opportunity for round the clock development facilitated by the temporal difference between remote development locations. The logic underpinning this approach is that these two factors can facilitate competitive pricing and reduce time to market. Thus enabling companies to compete more effectively by gaining, expanding or maintaining their market share [6].

As many organisations who have implemented a Global Software Development (GSD) strategy have discovered, due to the level of complexity involved in software development, outsourcing to other organisations or offshoring to remote divisions is not a straightforward task [3, 6-8]. Some of the difficulties encountered include such factors as the problem of understanding requirements, testing of systems and the coordination of these types of projects [7]. These difficulties are further compounded by cultural and language differences, lack of communication, geographical and temporal distance from

team members and the customer, different process maturity levels, development and testing tools, standards, technical ability and experience. As a result the management of globally distributed software development projects has been recognised as a difficult and complex task [9].

Given all these circumstances it is not surprising that offshoring and outsourcing software development has proved a complex endeavour and should never be embarked on lightly or without due consideration. A major problem which has emerged in this area is that too often the implementation of an outsourcing or offshoring strategy has been seen as simply the replication of those strategies which are implemented for collocated software development. This short sighted approach has led to serious problems and numerous failures [2, 7]. It is in this context and with the objective of helping to address the issues which have been outlined the authors have undertaken to develop the *GSD Implementation Model*.

2 Three Independent Case Studies

The findings presented in this paper are based on the results from three independent case studies which the authors have undertaken over an eight year period in the area of distributed software development. The first case study was carried out in an Irish company called Irish Computing Solutions (a pseudonym) who implemented a strategy to expand their organisation's market share by the establishment of local offsite virtual software development teams. Prior to implementing this policy the company operated collocated teams based in the capital (Dublin) who worked exclusively on the development of financial and telecommunications software. In addition the organisation had a software development centre located 150 miles from Dublin. This centre was involved in general application development and maintenance and had lower labour costs than the capital. The objective was to leverage staff at both locations and capitalize on the cost advantage which this strategy offered. A group of twelve offsite engineers were selected and were provided with basic training in the technology and process required. Two virtual teams were established and consisted of two sets of six offsite engineers who were partnered with three experienced onsite engineers based in Dublin. Considerable effort was put into providing the communication infrastructure, process and support for both virtual teams. A key objective of this approach was that the onsite engineers would mentor the inexperienced offsite staff and provide effective knowledge transfer. The operation of these teams and there subsequent failure provided the basis for this case study [10].

The second case study focused on what is termed offshore / nearshore software development [1]. The concept of offshore / nearshore is derived from the fact that the research centred on a partnership between a large US based financial organisation Stock Exchange Trading Inc. and an Irish division of a US multinational company Software Future Technologies (both pseudonyms). The US and Irish based sites were geographically distant, but they were considered linguistically and culturally nearshore [1, 11]. This partnership ultimately resulted in the establishment of virtual teams to develop and maintain bespoke financial software. Stock Exchange Trading Inc. was the senior partner in this relationship and had an on going requirement for the development and maintenance of this type of software. An unanticipated and urgent requirement arose for the development of new software during the initial stage of establishing the virtual teams. To address this need 70 percent of the Irish team members moved to the US, as a temporary measure for a period of one year to work on collocated teams with their Stock Exchange Trading colleagues. This proved to be a very effective strategy and both groups operated very successfully while collocated within what were to eventually become their virtual teams. It was only when the Irish team members returned to Ireland and the virtual teams were established that serious problems arose. These problems and issues and there ultimate solution have been articulated in detail in [10, 12, 13].

The third case study centred on offshore virtual team software testing and was undertaken in the Irish division of a large US multinational called Computing World International (a pseudonym) who had been operating in Ireland for over twenty years. The Irish division had been very successful and had expanded considerably over that time. During that period a large percentage of the projects undertaken had been offshored from their US parent; therefore, the Irish staff and management were very experienced in having projects offshored to them

Two years prior to undertaking this case study the organisation's corporate strategy changed. At that

time they initiated a policy of establishing virtual testing teams with the objective of leveraging the technical ability of their Irish staff with the competitive salary levels of their Malaysian test engineers. When this research commenced four virtual testing teams were in operation between the Irish and Malaysian divisions. Some teams were established for over a year and a half while others had only been in operation for a number of months.

This case study focused on two embedded units of analysis. One was a virtual testing team with members located in Ireland and Malaysia which had been in operation for a period of eighteen months. The second was a virtual team with a similar makeup, but had been established for just over six months. The different aspects and findings from this study have been outlined in detail and published in [10, 13-15].

2.1 Research Methodologies

The research methodology employed in the first and second case studies was the action research five-phase cyclical process based approach as defined by Susman and Evered [16] and Baskerville [17]. Action research entails the analysis of the direct intervention of the researcher. This methodology was selected as the most appropriate for both case studies as one of the authors held a management role in the respective organisations researched. The objective in both situations was to leverage the research opportunities which this provided while maintaining the required level of objectivity of both researchers. The third case study required a different approach and research methodology. When this study was undertaken both authors were fulltime researchers and were offered the opportunity to undertake extensive on site research. The objective was therefore to maximize the level of access this opportunity provided. After due consideration this resulted in the selection and implementation of a Yin [18] based embedded case study which incorporated a Strauss and Corbin grounded theory [19] approach to data gathering and analysis.

3 The Development of the GSD Implementation Model

Based on the analysis of the combined results from the three case studies [10, 12-15] ten key factors were identified. It was determined these factors were directly relevant and needed to be specifically addressed in order to establish and facilitate the operation of globally distributed virtual teams. These factors are summarised as follows:

1. Understand why, at what cost and risk a distributed strategy is undertaken
2. The Provision of effective infrastructure, process and documentation
3. The requirement to effectively establish the teams
4. Implement an efficient distributed team project management strategy
5. Ensure the development of common goals, objectives and rewards
6. The need for the clear definition of roles and responsibilities
7. Address issues related to culture, communication, motivation and fear
8. Ensure provision of adequate training and knowledge transfer
9. Facilitate and monitor the operation of collaborative and supportive teams
10. Document and leverage lessons learned

3.1 Foundation of the Model

Reviewing the ten key factors which were identified by this research it was determined of value to consider how they could be utilised to develop a strategy for the establishment, operation and the effective management of virtual software teams. It was realised they also had relevance and implications for GSD in general. To address both of these issues a model was developed which highlighted the key areas which needed to be considered and addressed to facilitate successful virtual team operation and globally distributed software development.

When developing this model it was recognised that it required to be clear so that it could be easily understood and implemented, to be practical so that it would be used and to be comprehensive to address the numerous relevant factors and issues which impact on GSD. It was also required to incorporate an element which facilitated recording relevant experience and knowledge gained while establishing and operating the GSD teams. This could then be leveraged to improve existing operations and assist with the implementation of GSD strategies in the future.

It was in this context that the IDEAL[sm] model [20] was researched and identified as an appropriate basis for the development of the *GSD Implementation Model*. The original focus and application of the IDEAL[sm] model is in the area of Software Process Improvement (SPI). In these circumstances the authors had in previous research utilised it as an effective tool and its adaptability had been successfully implemented to achieve SPI [21]. Its wider applicability and potential for use outside this specific SPI area has been recognised by the Software Engineering Institute (SEI). It is acknowledged that the model can provide an effective and disciplined approach for the adoption of new software engineering processes, methods and tools. In these circumstances it can also be utilised for establishing the foundation for and the maintenance of a long-term improvement strategy [22].

It was recognised that the IDEAL[sm] model presented a structure which could be amended to directly address all the relevant requirements and areas of concern which impact on the establishment and operation of GSD teams. It provided a simple, but comprehensive framework on which the *GSD Implementation Model* could be based. It also offered a straightforward, practical and extensive approach. Based on all these factors it was considered suitable. It has been adapted to the specific requirements of the GSD environment. What was proposed was not to mirror the IDEAL[sm] model in every aspect, but to utilise its relevant constituent parts and overall approach. Therefore the development of the GSD Implementation Model was based on the basic structure of the IDEAL[sm] model which was expanded and modified to meet the specific requirements and needs of operating in the globally distributed software development environment.

4 The GSD Implementation Model

The ten key factors which our research identified were divided into five distinct phases, which were to be undertaken sequentially. The model as a whole was designed for iterative execution (see figure 1). The five phases are as follows:

Initiating – Determine why, if and how the GSD approach is to be implemented

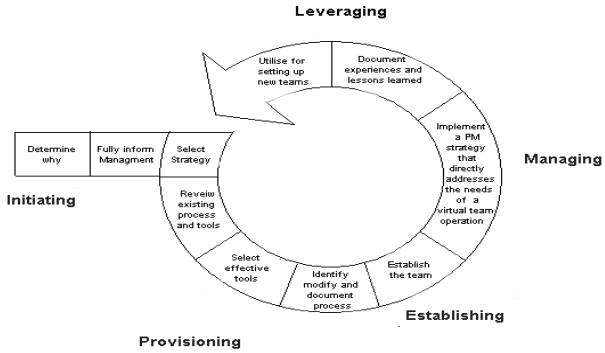
Provisioning – Ensure provision of effective infrastructure, process and documentation

Establishing –The requirement to effectively establish the GSD teams

Managing – Implementation of an efficient GSD project management strategy

Leveraging – Document and leverage lessons learned for existing and future projects

[sm] IDEAL is a service mark of Carnegie Mellon University



The GSD Implementation Model
Figure 1

4.1 Initiating

There is a requirement for organisations considering outsourcing or offshoring part or all of their software development activities to clearly define and articulate their rationale for selecting and implementing such an approach. In some cases justification is simply the result of a perceived cost advantage of implementing a GSD strategy at a corporate level or the fact that competitors are doing it. In a number of situations this type of rationale has proved very short sighted and led to serious problems. In these circumstances it is important that organisations recognise that the reality can be quite different. GSD projects can and have ended up costing as much or more than if they were collocated. They can also negatively impact on the delivery and quality of the software artefacts produced and the morale and motivation of existing staff [10, 13, 15].

In addition risk is a key factor which needs to be specifically addressed in the GSD environment, while pervasive risk should be incorporated into all well planned software projects [23, 24]. Globally distributed development projects carry additional high risk exposure [25]. These include the risk of delay or failure due to linguistic, cultural difference, motivation and temporal distance. All these issues need to be recognised and understood prior to embarking on or implementing such an approach [2, 26]. This can only take place when time is spent gathering and evaluating information on exactly what is involved and what are the positive and negative factors which are inherent to operating in a GSD environment.

If it is decided this is the strategy the organisation wishes to implement, the real potential costs and risks involved need to be accurately assessed. Based on these realistic projections the objectives of the strategy should be determined and directly linked to the short and long-term goals of the organisation. Senior management support is key to the success of any distributed software development strategy. Therefore, they must be provided with all the information necessary to allow them to have realistic expectations as to what can be actually achieved. Once the decision to implement this approach has been agreed the most appropriate GSD strategy should be selected.

4.2 Provisioning

Having selected a GSD strategy the infrastructure to support its implementation needs to be determined and put in place. In this context existing tools and processes need to be reviewed, adjusted and augmented. In some low cost locations the availability of a dependable electrical supply and al-

ternative power source need to be considered and addressed. Of equal importance is the availability of an adequate telecommunications infrastructure. Once basic infrastructure has been established across the relevant sites common or compatible tools need to be identified and sourced. This is required to ensure the interoperability of cross-site operations and artefacts. In this context an essential aspect of GSD is the selection and implementation of an effective configuration management system [7]. Due consideration also needs to be given to the selection of appropriate communication tools which are essential when operating in what can largely be an asynchronous environment [3, 12, 14].

Once adequate infrastructure is in place the identification and adoption of a common and effective GSD process needs to be considered [7]. Organisations must reassess and modify their existing processes for use in a distributed environment [26]. This includes the need for more formal methods of collaboration and communication given the loss of informal communication methods [27]. In the GSD situation there is a clear need for a well-defined jointly formulated and documented process to be put in place [21].

4.3 Establishing

The next step is to effectively establish the teams. Team members should be recruited internally and externally based on the technical needs of the project. Provision should be made for technical, cultural and communications training which are specific to the needs of the GSD environment [15]. The foundation for effective knowledge transfer between team members regardless of location should be put in place. This includes leveraging all visits between team sites to develop relationships. A priority from an initial stage is the establishment of a one-team vision and cooperative approach between team members regardless of location. This has to be actively fostered, developed and monitored[14].

4.4 Managing

There is the need for the development and implementation of an efficient GSD project management strategy which incorporates and addresses the specific requirements of operating in a distributed environment [14, 15]. In this context there is a need to facilitate and ensure the development of common goals, objectives and rewards. This is achieved by specifically addressing the issues, factors and variables that GSD teams are exposed to [3, 7]. There is also the requirement for roles and responsibilities to be clearly defined and articulated to all managers and team members. This is achieved through the use of a common vocabulary which unambiguously outlines this information.

There is also a requirement to address issues which are specifically related to culture, communication, motivation and fear [10]. This is achieved by understanding these issues and ensuring they are monitored and that timely and corrective action is taken to address any problems which arise due to any of these areas. Of equal importance is to monitor the effectiveness of technical training and knowledge transfer. When the requirement for additional training is identified it should be provided. If problems are identified with knowledge transfer they need to be investigated and specifically addressed. There should also be incentives to encourage staff to effectively transfer knowledge.

A cohesive team does not emerge of its own accord from a globally distributed, culturally, linguistically and technically diverse group of individuals, who are separated by geographical and temporal distance [7]. If it is to be put in place, it requires effort and goodwill on all sides. It can happen, but it must be planned, established, supported, monitored and actively developed. It can only take place with effective management where the positive aspects of the GSD environment are effectively leveraged and the negative factors and issues are addressed [12].

4.5 Leveraging

A key activity is leveraging the experience and knowledge gained by implementing a GSD strategy. This is best achieved by analysing and documenting the experience and knowledge gained. This should then be utilised to review what has been achieved and identify areas where further improve-

ments can be made. This information should also be made available and used to directly assist with the management of other existing teams and the establishment and operation of new GSD projects.

5 Conclusion

The *GSD Implementation Model* provides an overview which is practical and comprehensive in its structured and iterative approach. Within its five phases it addresses the specific requirements of operating in a GSD environment. This is achieved by ensuring the rationale for undertaking this approach is clearly articulated and understood and that realistic objectives and goals are set. Senior management support is secured on achievable expectations based on the accurate evaluation of costs and risks. The required infrastructure, processes and supports are put in place to facilitate the operation of the GSD teams. Time and effort is put into effectively establishing and managing the teams. An effective project management strategy based on the needs of the GSD environment is implemented. Key to the long term success of this approach is the documenting and leveraging of the experience gained implementing such a strategy. This model has been presented to forty five senior managers who had direct experience of implementing GSD strategies for evaluation. Their response was very positive and the consensus was that it was an excellent model to utilise when embarking on a GSD strategy as it highlighted the key areas which need to be specifically addressed.

6 Literature

1. Hayes, I.S., Ready or Not: Global Sourcing Is in Your IT Future. Cutter IT Journal, 2002. 15(11): p. 5 - 11.
2. Prikladnicki, R., J.L.N. Audy, and R. Evaristo, Global software development in practice lessons learned Software Process Improvement and Practice, 2003. 8(4): p. 267 - 279.
3. Karolak, D.W., Global Software Development: Managing Virtual Teams and Environments. 1999, Los Alamitos, CA, USA IEEE Computer Society Press.
4. Carmel, E. and R. Agarwal, Tactical Approaches for Alleviating Distance in Global Software Development. IEEE Software, 2001. 1(2): p. 22 - 29.
5. Crow, G. and B. Muthuswamy, International Outsourcing in the Information Technology Industry: Trends and Implications. Communications of the International Information Management Association, 2003 3(1): p. 25 - 34.
6. Herbsleb, J.D. and D. Moitra, Global Software Development. IEEE Software, 2001. 18(2): p. 16 - 20.
7. Carmel, E., Global Software Teams: Collaboration Across Borders and Time Zones. 1999, Saddle River, NJ: Prentice Hall.
8. Clerc, V., P. Lago, and H. van Vliet. Global Software Development: Are Architectural Rules the Answer? . in Second IEEE International Conference on Global Software Engineering, 2007. ICGSE 2007. 2007. Munich: IEEE.
9. Lanubile, F., D. Damian, and H.L. Oppenheimer, Global software development: technical, organizational, and social challenges SIGSOFT Software Engineering Notes, 2003. 28(6): p. 1 - 4.
10. Casey, V. and I. Richardson, The Impact of Fear on the Operation of Virtual Teams, in International Conference on Global Software Engineering, ICGSE 2008, IEEE: Bangalore, India.
11. Sahay, S., B. Nicholson, and S. Krishna, Global IT Outsourcing: Software Development across Borders. 2003, Cambridge UK: Cambridge University Press.
12. Casey, V. and I. Richardson. Practical Experience of Virtual Team Software Development. in Euro SPI 2004 European Software Process Improvement, . 2004. Trondheim, Norway.
13. Casey, V. and I. Richardson. Virtual Software Teams: Overcoming the Obstacles. in 3rd World Congress for Software Quality. 2005. Munich, Germany.
14. Casey, V. and I. Richardson. Uncovering the Reality within Virtual Software Teams. in First International Workshop on Global Software Development for the Practitioner, ICSE 2006 2006. Shanghai, China ACM Press

15. Casey, V. and I. Richardson. Project Management within Virtual Software Teams. in International Conference on Global Software Engineering, ICGSE 2006. 2006. Florianopolis, Brazil: IEEE.
16. Susman, G. and R. Evered, An Assessment of the Scientific Merits of Action Research, . The Administrative Science Quarterly, 1978. 23(4): p. 582 - 603.
17. Baskerville, R.L., Distinguishing Action Research from Paticipative Case Studies Journal of Systems and Information Technology, 1997. 1(1): p. 25 -45.
18. Yin, R.K., Case study research / design and methods Second ed. Applied Social Research Methods Vol. 5. 1994, Thousand Oaks, CA. USA: Sage Publications.
19. Strauss, A. and J. Corbin, Basics of Qualitative Research : Techniques and Procedures for Developing Grounded Theory. Second ed. 1998, Thousand Oaks, CA, USA: Sage Publications.
20. Paulk, M.C., et al., The Capability Maturity Model: Guidelines for Improving the Software Process. SFI Series in Software Engineering. 1997, Reading, MA. USA: Addison-Wesley.
21. Casey, V. and I. Richardson, A Practical Application of the IDEAL Model. Software Process Improvement and Practice, 2004. 9(3): p. 123 - 132.
22. Gremba, J. and C. Myers, The IDEAL Model: A Practical Guide for Improvement. Bridge, 1997(3).
23. van Vliet, H., Software Engineering : Principles and Practice. 2nd ed. 2000, Chichester, U.K. : Wiley.
24. Henry, J., Software Project Management: A Real-World Guide to Success 2003, Boston, MA, USA: Addison-Wesley.
25. Ebert, C., et al. Improving validation activities in a global software development in Proceedings of the 23rd International Conference on Software Engineering 2001 Toronto, Ontario, Canada IEEE Computer Society.
26. Nidiffer, K.E. and D. Dolan, Evolving distributed project management. IEEE Software 2005. 22(5): p. 63 - 72
27. Herbsleb, J.D. and R.E. Grinter, Architectures, coordination, and distance: Conway's law and beyond. IEEE Software, 1999. 16 (5): p. 63 - 70

7 Author CVs

Val Casey PhD

Dr. Val Casey is a researcher with Lero - the Irish Software Engineering Research Centre at the University of Limerick. His PhD research was carried out in the area of global software development. He has over 20 years experience in the IT industry and has also lectured in the University of Limerick. He is a SEI trained CMM assessor and holds a MSc. in Software Re-Engineering and a BSc. in Economics and Organisational Theory. His last industrial role was that of Quality Manager in an organisation which implemented a global software development strategy. He has also provided consultancy services focusing on software process improvement and software testing to the financial and telecom sectors.

Ita Richardson PhD

Dr. Ita Richardson is a senior lecturer in the Department of Computer Science and Information Systems at the University of Limerick. Her main research interests are Software Process Improvement with a specific focus on small to Medium Sized Enterprises (SMEs) and on Global Software Development. She is a project leader on the Global Software Development for SMEs project, which is funded by Science Foundation Ireland and operates within Lero - the Irish Software Engineering Research Centre. Her research involves qualitative research with companies, and some of her post-graduate students are in full-time employment with software development companies.

Experiences in Developing Requirements for a Clinical Laboratory Information System in Brazil

Jean Carlo R. Hauck[1,2], Maiara Heil Cancian[1], Christiane Gresse von Wangenheim[1, 2], Marcello Thiry[2], Aldo von Wangenheim[1], Richard H. de Souza[1]
[1]Federal University of Santa Catarina (UFSC)
Florianópolis/SC – Brazil
[2]Universidade do Vale do Itajaí (UNIVALI) – Computer Science
São José/SC – Brazil
jeanhauck@egc.ufsc.br, maiara@telemedicina.ufsc.br, gresse@gmail.com, marcello.thiry@gmail.com, awangenh@inf.ufsc.br, richardhenrique@gmail.com

Abstract

The development of health care information systems has been shown to be complex and costly. A primary concern is the thorough analysis of stakeholders' information, knowledge and their needs. Traditional processes have to be adapted in order maximize their efficiency and effectiveness. In this paper, we describe a requirements development process in alignment with CMMI-DEV and ISO/IEC 15504, which has been defined and applied for the development of new clinical laboratory information system for the Central Laboratory of the Santa Catarina State Public Health Department (LACEN)/Brazil. The system is currently being applied statewide for more than 75 types of clinical analyses involving 7 facilities. We present our experiences in applying the requirement development process and lessons learned.

Keywords

Requirements Analysis, Health Care, Clinical Laboratory Information System, Telemedicine

1 Introduction

Brazil is the largest economy in South America and its healthcare sector is considered to be worth approximately $56 billion per year [1]. IT in the healthcare sector has significantly advanced in the last years, as a consequence of both private and public investments in the sector. Web-based systems, for decentralized healthcare access, telemedicine and continuous education of health care personnel play a central role in this context.

But, still, only a minority of Brazilian hospitals and clinical analysis laboratories has some kind of integrated system solutions [2], and even fewer exist that are web-based or telemedicine capable. Therefore, there exists, among others, a critical need for efficient clinical laboratory information systems (CLIS) designed to store, manipulate, and retrieve information for planning, organizing, directing, and controlling administrative and activities associated with the provision and utilization of clinical laboratory services. And, to allow the prompt integration of external exam results into the hospital routine and to make them also accessible to clinicians and patients, it is extremely desirable to design a CLIS as part of a distributed e-health network.

Yet, the development of information system in the health care domain has shown to be complex and costly [3]. They have to be highly collaborative, involving a large number of different stakeholders, including, doctors, nurses, clerks, etc. Eliciting their needs is difficult. Often they do not even have

sufficient time to participate in the requirement development due to their typical working conditions, nor the expertise to discover errors or missing requirements [4]. On the other side, requirement development in the health care domain, requires a thorough understanding of the domain specific concepts and terminology [5]. There exist also a large number of unconscious requirements, which may simply be overlooked [4]. Typically, health care information processes are also characterized by a large number of exceptions [6], difficult to elicit, as well as non-functional requirements, which are numerous and important, especially regarding safety and security [5]. The workflows in place may not be optimal, requiring innovation as part of the deployment of an information system [7]. Generally, health care information systems also must interface with instruments or other information systems. Therefore, a primary concern is to organize the development of health care information systems in such a way to maximize their efficiency and probability of success [6]. Of critical importance is the thorough analysis of stakeholders' information, knowledge and their requirements [7]. In this context, traditional requirements development processes have to be adapted to be effective and efficient [5].

In this scenario, the CYCLOPS Group [8] at the Federal University of Santa Catarina/Brazil aims at the development and transfer of innovative methods, techniques and tools in the health care domain, including telemedicine, medical image analysis, 3D imaging in cooperation with several hospitals and medical clinics. Recognizing the need to improve its software process, the CYCLOPS Group started an improvement program in 2006. Its software process has been organized and modeled in accordance to CMMI-DEV [9], ISO/IEC 15504 [10] and MPS.BR [11] with a special focus on the development and management of requirements (Figure 1).

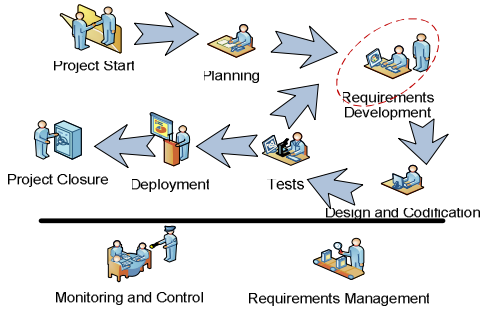


Figure 1. Overview of the CYCLOPS software process

In this paper, we describe our experiences regarding the development of software requirements for a CLIS at the Central Laboratory of the Santa Catarina State Public Health Department (LACEN). LACEN is a public institution that provides services for the state`s public healthcare network, which is part of the SUS – *Sistema Unico de Saúde*, the Brazilian public single payer health system. It has a central facility located at the capital of the Santa Catarina State and 6 facilities distributed across the state. Today, LACEN executes about 20.000 clinical analyses per month, which it receives from various institutions, such as public and private hospitals, primary healthcare facilities, etc.

Figure 2. Receiving a blood sample at LACEN

The objective of the project is to develop a web-based clinical laboratory information system supporting patient order entry, specimen processing, result(s) entry, entry tracking, web-based results inquiry, preliminary, final and public health reporting and patient demographics. The system also provides HL7-compliant interfaces to reference labs and Electronic Health Records (EHRs). The system will be applied for more than 75 types of laboratorial analyses in 25 areas, such as, hematology, immunology, etc. One important requirement was that the system was to be designed to be integrated into the Santa Catarina State Telemedicine Network (RCTM) [12], enabling it to feed the statewide public EHR system being developed in this context. The project started in July 2006, and, currently, the system is being deployed.

2 Requirements Development

One of the critical challenges in the project was the development of the requirements to assure the development of a software system that actually meets the user needs. Here, we use the term "requirements development" to refer to the process to elicit, gather, model, specify, analyze, validate, document and communicate data, information and requirements that are needed to support the respective business process. As part of the software process improvement program at the CYCLOPS Group, the current requirements development process in place was described and improved in alignment with the principal reference models, including CMMI-DEV, ISO/IEC 15504 and MPS.BR. Figure 3 illustrates the basic steps of the requirements development process.

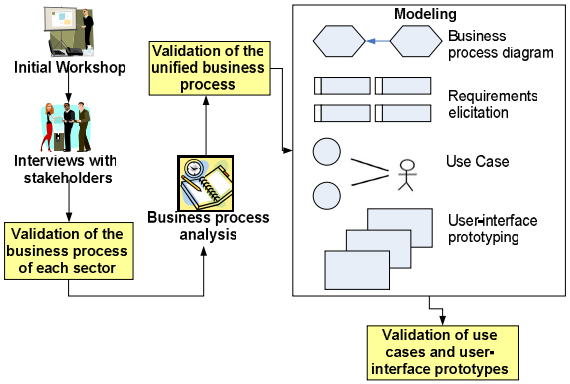


Figure 3. Overview of the requirements development process

In the LACEN project, we applied the defined process as follows:

Initial Workshop. First, we organized a 3-hours workshop involving all stakeholders in order to establish a common understanding on the scope of the project and to obtain their commitment. During the workshop, the project manager presented the CYCLOPS group and the objective and scope of the project. The director of the LACEN presented an overview on the institution, its objective and organizational structure. Then, one representative of each sector of the LACEN described the sector and presented a high-level workflow overview. These presentations had been prepared in advance together with software analysts of the CYCLOPS Group. The presentations helped to understand the context of the system to be developed and to identify relevant stakeholders. At the end of the workshop, the sponsors emphasized once more the importance of the project and their support, motivating the involvement of the stakeholders in the project. The whole workshop was filmed and the information obtained was documented by the software analysts.

Interviews with stakeholders. Based on the information obtained in the workshop, we decided to perform a group interview with relevant stakeholders per sector. A schedule for the interviews was developed and revised with LACENs' management verifying the availability of respective stakeholders.

Then, collaborative interviews were conducted using an adaptation of JAD sessions [13]. The objective of these interviews was to elicit the workflow executed by each of the sectors. In addition, we analyzed artefacts being consumed or produced as well as the inter-relationship of the sectors' process with other processes of the organization. Each interview has been analyzed by describing the respective business process in a textual form as well as a graphical model. For this visualization, we used simple stereotypes related to the health care domain, in order to facilitate the understanding of the process by the stakeholders (Figure 4).

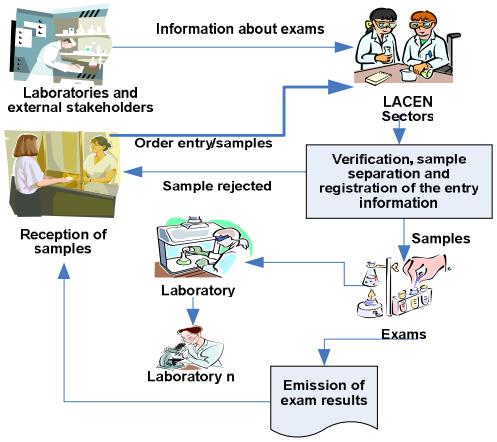


Figure 4. Sample extract of business process of LACEN

During the interviews, also a vocabulary of the application domain was elicited, creating a glossary as part of the requirements document.

Validation of the business process and vocabulary of each sector. We organized validation sessions with 2-3 representatives of each sector of the LACEN during which we step-by-step presented and discussed the process, as elicited in the interviews. Errors or missing information were corrected until we obtained a version formally approved by the stakeholders.

Business process analysis. One of the objectives of the project was to establish a system for an integrated and uniform business process among all sectors of LACEN. Therefore, we included a step in which we analyzed and unified the elicited business processes of each sector. Inconsistencies were solved in cooperation with representatives of the respective sectors and senior management of the LACEN.

Validation of the unified business process. We validated the unified business process with senior management of the LACEN. Therefore, the unified business process was presented in a validation session and discussed. In the end, the process was formally approved by senior management.

Modelling of the business process diagram. Then, we modelled the business process more formally, adopting a customized notation based on UML [13] using Enterprise Architect (EA) [14] (Figure 5). Besides the formalization of the business process, this activity also served to understand in more detail the structure and dynamic of the institution's processes. At this moment, we focused on understanding the processes in place, without the usage of an information system. The detailed documentation of the business process permitted its systematic analysis and the identification of improvement opportunities.

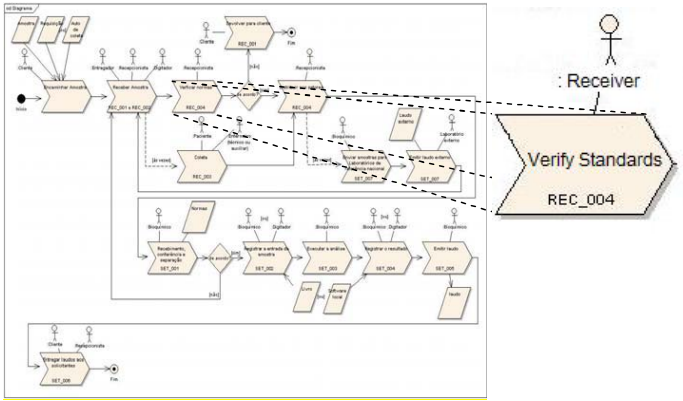


Figure 5. Sample of the business process of LACEN

Requirements elicitation. Based on the modelled business process, we begun to elicit functional and non-functional requirements of the system to be developed, considering the identified improvement opportunities. Requirements and business rules were documented in a textual form using a customized template of EA.

Use case development. Then, we grouped the functional requirements per functionality and started to develop use cases based on the information elicited earlier. Once all requirements were mapped to use cases, a use case diagram was developed providing an overview on the scope of the system. All use cases were initially described high-level and, later on, refined detailing the main flow, alternatives and exceptions.

User-interface prototyping. In parallel to the refinement of the use cases, we also prototyped the user-interfaces of the system in accordance to the event flows defined in the use cases. The objective of the prototyping was to facilitate the validation of the future system by the stakeholders.

Validation of use cases and user-interface prototypes. This has been done in two steps. First internally by a validation of software process engineers of the CYCLOPS Group throughout the whole development process, analyzing the conformance of the process and artefacts with respect to the defined process as well as the correctness and consistency of the requirements, business rules, use cases and user-interface prototypes. In a second step, the use cases and user-interface prototypes have been presented to the principal representatives of each sector and senior management of the LACEN and their formal approval was obtained after some small adjustments.

As result of the requirement development a formally approved requirements document was completed, which served as input to the second stage of the project, covering design, codification, tests and deployment of the system.

3 Lessons Learned

In general, the requirements development process as described in Section 2 worked well in the LACEN project. Applying the process, we observed various lessons learned:

Capturing the "big picture". Organizing the initial workshop, was not sufficient to assure that a general view and context of the system to be developed was captured. Therefore, we are including an additional step after the workshop, involving a more thorough characterization of the institution(s) in which the system will be implanted as well as profiles of its future users. We also added requirement elicitation sessions with senior management and sponsors of the respective institution(s).

Group interviews. A specific characteristic of the LACEN project was the prior inexistence of a de-

fined unified process. In this context, the realization of group interviews allowed to discuss and solve inconsistencies or conflicts immediately with the involved parties. Only in cases where conflicts could not be resolved directly, senior management had to be involved together with the representatives of the respective sector(s).

Involvement of senior management. Their involvement, when necessary, and their consistent way of decision and motivation for the unification of the processes has been essential for the definition of a uniform processes to which all sectors are committed.

Non-functional requirements. Our approach has shown to be adequate for the elicitation of functional requirements related to the actual business process in place. However, we also observed two shortcomings. The approach did not sufficiently help to elicit non-functional requirements – a typical difficulty in this domain [5]. Therefore, we included a specific step for the elicitation of non-functional requirements using a checklist based on [15].

Innovation. We also experienced difficulties in modelling a system that meet users' needs regarding the actual process in place while at the same time proposing an innovative technological solution. Therefore, we are also integrating the consultation of technology experts in the health care domain to complete the requirements development.

Documentation. In general, the format and contents of the requirements documentation was adequate, but we observed the need to also document the relationship (dependency) between requirements, a data dictionary for user-interfaces as well as the navigation flow between interfaces.

Step-wise validation. Realizing first the validation of the business process and later on the validation of the use case and user-interface prototypes, was experienced as beneficial as it helped to identify errors early and concentrated the involvement of the users to the validation of parts of the processes of their sector. We also experienced the realization of validation meetings with an analyst and users as essential. It would not have worked, if we had simply asked the representatives to review the documents individually.

Formal approval. During the later steps in the requirements development and especially during the implementation of the system, we experienced the formal approvals we obtained as essential for negotiation each time a change request came up with significant impact on the project baseline.

Involvement of future users. We also experienced the close involvement of future users as one of the key factors of success. The initial workshop helped to provide an understanding of the project and to create a friendly climate. The active participation during the elicitation and validation contributed to obtain a better understanding of the requirements.

4 Conclusion

In this paper, we present our experiences in developing requirements for a new CLIS. The project has also served as a pilot for the deployment of a defined requirements development process in alignment with CMMI-DEV, ISO/IEC 15504 and MPS.BR for the development of new systems in the CYCLOPS group. In order to establish the requirements development process at the CYCLOPS group, we used the ASPE-MSC approach [16], which, based on the descriptive modelling of the process in place, identifies strengths and weaknesses and guides the improvement in alignment with reference models, such as, CMMI or ISO/IEC 15504. Our experiences show that such a hybrid process improvement approach, using descriptive and prescriptive techniques, facilitates SPI by strengthening the existing process and culture, instead of trying to adopt a generic process top-down. We experienced the objectives/purpose and practices/outcomes as defined with regard to the requirements process by CMMI-DEV, ISO/IEC 15504 and MPS.BR as adequate. All three models provide a consistent view, which is defined on a level sufficiently abstract to be tailorable to a specific environment. On the other side, we noticed a lack of more concrete guidance, which maps practices/outcomes of reference models or standards to alternative processes, techniques and/or tools to satisfy these requirements.

Acknowledgements

Our thanks to all involved in the project at the LACEN, the Santa Catarina State Health Department and our colleagues at the CYCLOPS group/UFSC and LQPS/UNIVALI for their support.

Literature

1. US Fed News Service, April 1, 2006.
2. CS Brazil, Market Research: Software for the Healthcare Industry, April 2005.
3. Reddy, M. et al. Sociotechnical Requirements Analysis for Clinical Systems. Methods Inf Med., vol. 42, 4/ 2003.
4. Tveito, A., Hasvold, P. Requirements in the Medical Domain: Experiences and Prescriptions. IEEE Software, vol. 19, no. 6, Nov/Dec, 2002.
5. Cysneiros, L. M.. Requirements Engineering in the Health Care Domain. Proc. of the IEEE Joint Int. Conference on Requirements Engineering, Essen, Germany, 2002.
6. Reddy, M. et al. Sociotechnical Requirements Analysis for Clinical Systems. Methods Inf Med., vol. 42, 4/ 2003.
7. Ashry, N. Y., Taylor, W. A. Requirements Analysis as Innovation Diffusion. Proc.of the 33rd Hawaii Int. Conference on System Sciences, Hawaii, 2000.
8. CYCLOPS Group (http://cyclops.telemedicina.ufsc.br)
9. SEI. Capability Maturity Model Integration (CMMI) (http://www.sei.cmu.edu/cmmi)
10. ISO/IEC 15504 Information Technology - Process Assessment (Part 1 – Part 5), 2003 – 2006.
11. SOFTEX. Brazilian Software Process Improvement Model MPS.BR. (http://www.softex.br/mpsbr)
12. Maia, R. S., von Wangenheim, A., Nobre, L. F. A Statewide Telemedicine Network for Public Health in Brazil. Proc. of the 19th IEEE Symposium on Computer Based Medical Systems, Salt Lake City, 2006.
13. Thiry, M. et al. Uma Abordagem para a Modelagem Colaborativa de Processos de Software em Micro e Pequenas Empresas. Simpósio Brasileiro de Qualidade de Software, Brazil, 2006.
14. Enterprise Architect (http://www.sparxsystems.com.au/ea.htm)
15. Sociedade Brasileira de Informática em Saúde. Manual de Requisitos de Segurança, Conteúdo e Funcionalidades para Sistemas de Registro Eletrônico em Saúde, Fev 2004.
16. WANGENHEIM, C. G.; WEBER, S.; HAUCK, J. C.; TRENTIN, G. Experiences on Establishing Software Processes in Small Companies. Information and Software Technology, v. 48, n. 9, 2006.

Authors' CVs

Jean Carlo Rossa Hauck

Jean Carlo Rossa Hauck is SEPG manager of the CYCLOPS Research Group at the Federal University of Santa Catarina (UFSC). His research interests are in software process improvement and project management. He received his M.Sc. in Computer Science from the Universidade Federal de Santa Catarina and is a PhD student of the Graduate Program in Knowledge Engineering and Management at the Federal University of Santa Catarina. Contact him at UFSC - EGC, Campus Universitário 88049-200 Florianópolis/SC, Brazil; jean-hauck@egc.ufsc.br

Maiara Heil Cancian

Maiara Heil Cancian is system analyst of the CYCLOPS Research Group at the Federal University of Santa Catarina (UFSC). Her research interests are software process improvement and project management. She received his B.Sc. in Computer Science from the Universidade do Vale do Itajaí (UNIVALI) and is a master student of the Graduate Program in Automation and Systems at the Federal University of Santa Catarina. Contact her at UFSC -CTC-DAS, Campus Universitário 88049-200 Florianópolis/SC, Brazil; maiara@telemedicina.ufsc.br

Christiane Gresse von Wangenheim

Christiane Gresse von Wangenheim is a professor at the Universidade do Vale do Itajaí (UNIVALI) and consultant at Incremental Tecnologia. Her research interests are software process improvement, including project management. 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 (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. She's a member of the IEEE Computer Society, the Project Management Institute, and the Working Group ISO/IEC JTC1/SC7/WG24—SE Life-Cycle Profiles for Very Small Enterprises. Contact her at UNIVALI, Rod. SC 407, Km 04, 88122-000 São José/SC, Brazil; gresse@gmail.com

Aldo von Wangenheim

Aldo von Wangenheim is a Professor for Medical Informatics and Telemedicine at the University Hospital of Federal University of Santa Catarina and his research interests are in the areas of medical image analysis for diagnosis support and large-scale telemedicine frameworks for public health. He studied Computer Sciences at the Federal University of Santa Catarina – UFSC in Brazil and obtained his Ph.D. in Computer Sciences from the Industrial Mathematics Ph.D. Program at the University of Kaiserslautern, Germany. Contact him at UFSC -CTC-INE, Campus Universitário 88049-200 Florianópolis/SC, Brazil; awangenh@inf.ufsc.br

Richard H. de Souza

Richard H. de Souza is a master student of the Graduate Program in Computer Science at the Federal University of Santa Catarina. His research interests are in software process improvement and requirement management. He received his B.Sc. in Computer Science from the Universidade do Vale do Itajaí (UNIVALI). Contact him at UFSC -CTC-INE, Campus Universitário 88049-200 Florianópolis/SC, Brazil; richardhenrique@gmail.com

Reducing the Risk of Misalignment between Software Process Improvement Initiatives and Stakeholder Values

Nilay Oza[1], Stefan Biffl[2], Christian Frühwirth[1], Yana Selioukova[1], Riku Sarapisto[3]
[1]Helsinki University of Technology, Software Business Lab, Finland 02150
[2]Vienna University of Technology, Institute of Software Technology, Vienna, A-1040, Austria
[3]Basware Oyj, Software Product Company, Finland
{Nilay.Oza, Christian.Fruehwirth, Yana. Selioukova}@tkk.fi,
Stefan.Biffl@tuwien.ac.at, Riku.Sarapisto@basware.com

Abstract

Software companies who want to improve software process capabilities (SPCs) need a systematic method to make informed investment decisions on software process improvement (SPI) initiatives. Such decisions should aim at creating maximum stakeholder values. Existing software process assessment models such as CMMI and SPICE give an overview on relevant SPCs but do not help in aligning the investment decisions on SPI initiatives with stakeholder values. This echoes in our research experiences with the Finnish software companies. To address this problem, we present a method with tool support that may help companies align stakeholder values with SPCs and SPI initiatives. The proposed method has been developed based on the well-established "Quality Function Deployment" (QFD) approach. We report on experiences from applying the method and tool support in project workshops with SPI specialists and a Finnish software company. The experience with the proposed method suggests that it particularly helps to reduce the risk of misalignment by identifying those SPI initiatives that are most beneficial to stakeholders. The tool support provided with the proposed method also generated positive experiences in increasing the usability of the method and helped companies in the elicitation and prioritization of stakeholder values.

Keywords

Software Process Improvement, Risk Management, QFD.

1 Introduction

Company executives need to invest in change initiatives that are most likely to improve those core capabilities of the company that have considerable impact on benefits provided to customers and other success-critical stakeholders. Change initiatives include, but are not limited to, SPI initiatives and aim at improving a company's performance in delivering stakeholder values. Software quality teams often struggle to convince senior management to grant funding for SPI programs for lack of getting a clear picture of tangible benefits [9]. Even if there is common understanding on needed investment in SPI programs, senior management and the SPI team still may fail to invest in "right" capabilities, i.e., capabilities that best improve the value to stakeholders. By investing in right capabilities companies may diminish risk of spending financial assets on change initiatives that do not provide evident advantages to stakeholders.

While there are many potential benefits of SPI initiatives, one of the major risks is to focus on

initiatives that have only marginal effects on capabilities of the company and bottom-line benefits. We refer to such misleaded focus as "misalignment of SPI initiatives and stakeholder value". Software process assessment models, such as CMMI or Spice, are useful to give an overview on relevant target candidates in software process areas (SPAs) and provide feedback on process maturity to motivate SPI initiatives [3]. However, research work conducted at Software Business Lab (SBL)[1] through annual software industry surveys (OSKARI) [2] , and several project-specific industry collaborations with Finnish software companies, found a) only little overlap between CMMI/Spice SPAs and the companies' needs for improvements of core capabilities, b) considerable risk of investment into SPI initiatives that provide little tangible benefits to company stakeholders, and c) very little method and tool support, which can be used with experts from different domains (e.g., business and IT) in the company, for the evaluation and alignment of SPI initiatives. Based on these practical insights, in this article, we present a method for eliciting and aligning stakeholder values with a company's software process capabilities to identify the most promising SPI initiatives. Stakeholder value is the part of value-based requirement engineering activities which includes: *"identification of success-critical stakeholders; eliciting their value propositions with respect to the system; and reconciling their value propositions into a mutually satisfactory set of objectives for the system"* [2]. The proposed method is largely based on "Quality Function Deployment" (QFD) principles and is supported with a prototype tool for more efficient data collection and analysis. The method comprises two iterations:

I) The first iteration helps to understand the alignment/impact between stakeholder values and SPCs,

II) The second iteration helps to understand the alignment between SPCs and SPI initiatives.

Based on the analysis of alignment data from both iterations, the decision makers are more likely to make an informed decision on investing in "right" capabilities, which shows a strong connection between SPI initiatives, SPCs, and stakeholder values. Furthermore, the accompanying tool support intrinsically fosters an improved common understanding between senior management and SPI teams on the value of SPI initiatives. The main contribution of the proposed method is in its applicability in the real-world context. We report on initial experiences of executing the method and tool in a pilot workshop and a Finnish software company. The method and tool are undergoing further empirical examinations in Finnish software companies and SPI specialists in the research project - VASPO[3].

The remainder of this paper is structured as follows. Section 2 summarizes related work on stakeholder values, SPCs, SPI, and QFD and presents the main research focus. Section 3 presents the method and illustrates its steps. Section 4 presents and discusses experiences on the application of the method and a few limitations of the method. Section 5 concludes the paper with directions for further research.

2 Related work and research issues

There has been no dearth of software process models and improvement programs. Although all have some specific merits and argumentation, many of them have failed to convince the practitioners on the models' practical use. Among these, SPI programs have been most talked about, whether it is just "prestigious" to quote company's investment in SPI programs or "self satisfying" with continuous improvement. It is quite difficult for senior managers to see the "measurable" benefits, the problem addressed by a few scholars [4], [12]. However, here, we are addressing another practical concern of the practitioner that has been hardly addressed by SPI research – how to best align stakeholder value propositions against SPCs and SPI initiatives. Our proposition, in the form of an applicable method, is mainly based on research in the Finnish software industry. The next subsection captures the relevant background issues related to the two iterations of the proposed method.

[1] www.sbl.tkk.fi
[2] http://www.sbl.tkk.fi/oskari/index.htm
[3] The research work is carried out under Finnish national research agency – Tekes' funded project referred to as "Value-based Software Process and Organizational Change Management" (VASPO). More information about the VASPO project is available at www.vaspo.fi

2.1 Stakeholder value and software process capabilities (SPCs)

SPI initiatives are waste of money and energy if they are not useful to stakeholders. Stakeholders could be customers, programmers, executives or others who are in the scope of the SPI program. Stakeholder value is often depicted by success factors of SPI. Here, we distinguish stakeholder value with added emphasis on usefulness of offerings of SPI to its key beneficiaries. This is in line with Boehm [2] who argues that main failures in software projects are caused by value-oriented shortcomings since projects often fail to consider stakeholder values. The stakeholder values may range from user involvement, clear statement of requirements to executive management support, and proper planning depending on the stakeholder [3]. If we take examples of SP assessment models such as CMMI and Spice, we see little overlap between process areas of different such models. As a result they are limited in improving company's SPI capabilities. Within CMMI, for example, the SPAs in lower level of maturity ladder are well aligned to creating tangible stakeholder value such as software project planning, requirements management, software quality assurance, and software configuration management. CMMI and the related material proves a good cook-book of SPAs for practitioners; where to start, which order to execute, how to assess progress etc. But when it comes to measurement of increase of value produced, there seems to be a discontinuation between levels 1-3 and 4-5. On the lower levels it is evident that focusing on basic project management (incl. requirement management, configuration management, project planning, measurement and analysis etc) and further focusing on process standardization (incl. decision analysis and resolution, risk management, organizational areas, verification and validation, etc.) is likely to produce a significant improvement in both productivity and quality. However, at higher maturity levels, the alignment of SPAs seems to be weaker. For example process change management and technology change management are not well aligned. Nonetheless, beyond process models, stakeholder values also need proper alignment with SPCs.

The development of SPCs to address the change and organizational inertia must be done somewhere outside the scope of CMMI. On the higher CMM levels how can the SPI actions be valued? The issues will get very hard to understand and SPI initiatives hard to justify for people whose expertise is in the business disciplines instead of software engineering disciplines. A simple, clear and informative method together with a tool for data collection, analysis, and visualization is required i.e. for almost daily use to remind which initiatives seem most important, why and how they are selected. If we consider customers, a company should focus on "what capabilities do we need to develop for adding value for the customer" and view customers' strategic behaviour as a process of identifying and developing SP capabilities that company employs to "create unique level of value for selected customers and other stakeholders" [8]. This indeed requires multiple groups' involvement within the company to develop common understanding of stakeholder needs and values which in turns develops capabilities and competitive advantage [10]. Consequently, the alignment of stakeholder values and SPCs is essential for selecting the SPCs that become "right" candidates for SPI initiatives.

In the proposed method, we give opportunity to stakeholders to brainstorm and decide on aggregated consensus about the values they want to focus on when aligning with SPI initiatives, attempting to address the need of an informative method. This is captured under iteration I of the proposed method. In the next sub section we look at the relevant background behind iteration II, i.e., aligning SPCs and SPI initiatives.

2.2 Software process capabilities (SPCs) and SPI initiatives

Aligning stakeholder values with SPCs is not enough. There is considerable risk of investment into SPI initiatives that provide little tangible benefits to company stakeholders This is likely to happen if there is not proper alignment between SPI initiatives and relevant SPCs.

The standard SP and SPI models such as CMMI, SPICE etc. rightly address the SPC issue. If used effectively, they help identify key SPCs. However, they seem to be limited in helping to identify SPCs that are "right" for the company to address and invest in. Subsequently, often companies invest money and effort in "less important" capabilities and fail to capitalize on the key SPCs. Some of the critiques on SPI models show that assessments could be "wasteful". For example, Fayad and Laitnen [5] argue that CMM's practices are idealized practices and the real-world organization compares their practices

with an artificial list of practices that are just "ideal" and therefore some practices may not fit the "assessment criteria". It is also noted in our experience and the literature that business value from SPI programs is difficult to assess due to very few data points about the actual costs of such programs quoted by Fayad and Laitnen [5] as "*tantalizing possibilities rather than established facts"*. We, rather than reflecting on worthiness of SPI costs or SPI models, address the issue of aligning the investment in SPCs that need improvement. Particularly, lack of systematic method and tool support makes the alignment even harder. We address this issue in our method under iteration II. We execute overall method by proposing a practical tool based on Quality Function Deployment (QFD) approach. In the next subsection we summarize our use and position on QFD.

2.3 Quality function deployment (QFD)

We have used QFD to develop tool support for the proposed method as the QFD helps to concisely structure communications and link together information [6], similar to the structure the alignment challenge posed above. Subsequently, QFD can help in aggregating consensus-based alignment on SPI initiatives. QFD is a comprehensive method that takes into account stakeholder interests within the whole company horizontally and vertically, reflects market demand by using various types of specifications and production process variables which leads to production operations planning [6]. QFD application in software industry is not extensively reported by scholars; however several papers add valuable content to the body of knowledge in this domain. For example, Richardson et al. [11] offered a QFD-based method for SPI in small companies built on self assessment of software processes. Liu et al. [7] used software QFD (software domain adapted) as an approach to connect business goals and CMMI maturity levels by prioritizing requirements from multiple perspectives. The increased use of QFD method in practice has yet not been sufficiently facilitated by efficient tool support. Here, we have compromised some functions in tool support from the original QFD method to make it practically more usable in workshops with company executives. For example, QFD's requirement on capturing full feature set seems too time consuming and complex in scenarios with multiple stakeholders from different domains, like SPI initiative investment projects usually are. The tool addresses this complexity constraint through reducing the scope of the QFD method to its core feature, the alignment matrix and the planning matrix, which indeed suits our method very well.

2.4 Research focus

Evident from the argumentation in this article so far, the main research issue gathers around alignment of 1) Stakeholder values and SPCs and 2) SPCs and SPI initiatives. In this article, we are addressing these issues by presenting a method with tool support for proposed alignment and particularly experience of executing the method in the commercial setting. In the next section, we present the alignment method with a practical example.

3 Method description

In this section we explain and illustrate all steps of the proposed method, which is used in a workshop in which the SPI stakeholders participate. The method consists of seven sequential steps and is divided in two iterations. Figure 1 shows an overview of both iterations. On the left, it shows alignment matrix of stakeholder values and company's SPCs, that is an outcome of iteration I of the method. On the right, it shows the alignment matrix of SPCs and SPI initiatives, that is an outcome of iteration II. Figure 1 also depicts that the SPCs identified in iteration I are used in iteration II to map them against candidates for SPI initiatives. The impact symbols shown on the figure represent the degree of impact between row and column entities. Here, we describe all seven steps of two iterations detail. Figure 1 also shows the synopsis of the steps. The screenshots (based on the tool support) for both iterations are presented in Appendix 1.[4]

[4] For detailed worked out example with screenshots, please refer to www.vaspo.org/alignmenttool

Iteration I: Aligning stakeholder values with SPC

1. **Elicit stakeholder values:** The stakeholders are presented with a list of known value propositions and asked to expand the list by adding values which are specific to their company´s context. The compiled list then undergoes a two-staged reduction process (similar to the established in EasyWinWin approach [1]): firstly, each participant votes for the stakeholder value propositions to identify the 10 highest voted factors. Secondly, group negotiation, supported by a moderator and a software tool, prioritizes the identified top 10 factors.
2. **Organizational planning:** Stakeholders rate their company´s current performance in delivering each of the stakeholder values identified in step 1. They further set the planned target level for a particular value that should be achieved through one if the improvement initiatives. Stakeholders have the option to include external benchmarks or competitor ratings to set new targets. The performance rate uses a scale from 0 to 5, similar to SPICE´s levels, where 0 refers to the company not meeting the stakeholder value at all and 5 refers to fully delivering the value.

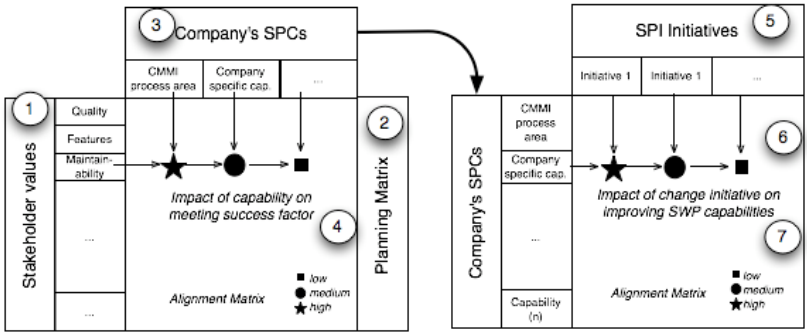


Figure 1: Overview of the method for aligning stakeholder values, software process capabilities, and SPI initiatives.

3. **Elicit company's SPCs:** The method provides a list of SPCs from the CMMI process areas by default. Like in Step 1, stakeholders are again asked to expand and edit the list with their company-specific SPC. Through this expansion process the method enables the stakeholders to combine the advantages of the more generic CMMI framework with their own more specific context. Similar to step 1, the expansion is followed by reduction through voting and prioritization.
4. **Align SPCs with stakeholder values:** Extracting results from step 1 to 3, a matrix is presented with stakeholder values as rows, and SPC as columns. The cells in the matrix are rated with strong, medium or low alignment symbols. The symbols refer to the impact of a SPC on the company's ability to meet a particular stakeholder value. The process of filling the matrix with impact symbols follows the previous prioritization of the stakeholder values and capabilities, with the most important items tackled first. Based on the impact symbols, cumulative impact of SPCs are calculated for each stakeholder value. The SPCs are then prioritized accordingly. The prioritized SPC list is referred to as "chosen SPCs".

Iteration II: Aligning SPCs with SPI initiatives

5. **Elicit SPI initiatives:** The stakeholders are provided with default set of SPI initiatives by the software tool and are asked to expand them with company specific initiatives. Like in previous steps (1 and 3), the expansion of the list is followed by a reduction.
6. **Align SPI initiatives with chosen SPCs:** SPI initiatives identified in step 5 are arranged in matrix form against SPCs identified in step 3. This gives the opportunity to analyze impact of SPI

initiatives on improving the SPCs. The stakeholders use the same impact symbols as in step 4 to fill the matrix, starting from the most important SPI initiatives and SPC.

7. **Choose the SPI Initiatives which have the strongest impact on improving chosen SPCs:** The first matrix (step 4) helps decision makers to determine which SPCs have the strongest impact on stakeholder value. We refer to these capabilities as "chosen SPCs". The second matrix (step 6) presents SPI initiatives and their impact on improving the "chosen SPCs". SPI initiatives with the strongest impact on the "chosen SPCs" are then considered best candidates for SPI investment. Different patterns of impact on SPCs from an SPI initiative are evaluated from the matrix before making the decision on investment.

3.1 Tool support

The tool support for the method provides an opportunity to apply it in the practical contexts. The proposed tool is developed using the spreadsheet metaphor. In the practical scenario, a moderator takes control of the tool and executes it based on given instructions (similar to EasyWinWin [1]). The moderator feeds the needed data to the tool based on participants' inputs. The tool support allows participants to focus more on the substance as all data arrangement and matrix development is taken care of by the tool. For example, the tool prioritizes the value factor list based on agreements negotiated between stakeholders and also visualizes the level of agreement for all stakeholders and allows the moderator to direct the argument to those items where consensus has not been reached. Subsequently, participants obtain the results immediately after the alignment process. Results in a matrix style systematic presentation form also aids further analysis of the results. For this paper, we have documented more detailed information on the tool, available at www.vaspo.org/alignmenttool/

4 Experiences from method application and limitations

In this section, we present our initial experience with the application of the method. Until now, we have implemented the method in workshops at two instances. We also report and discuss limitations of the method. Please note that due to confidentiality of company strategy the reported data and analysis results are exemplified.

4.1 A workshop with software process experts

In the workshop in the VASPO context stakeholder participated from three Finnish universities, three Finland-based software companies, and a member from Finnish government agency. This was useful for observing implementation of the method in a mixed stakeholder group. Promising results (based on the collected feedback from the workshop) of the method's application were:

- Six members were familiar with QFD style approach but had not used a practical tool.
- All the industry representatives showed interest in tailoring such a workshop for their company-specific context for better alignment between stakeholder values and SPI initiatives.
- Workshop participants found the method useful in real contexts, of practical value and easy to use.
- The notion of value should be cleared upfront to avoid possible different perception of stakeholder value.
- We made participants work in teams of 2 members. We found this setting generated more discussion and diverse perspectives in reaching the consensus to a particular point. This might not have happened if participants were working alone in the workshop.

4.2 A workshop with the Finnish software company

In the company workshop stakeholders participated from senior management, product management, and the quality assurance team. Promising results of the method's application were:

- Workshop participants found the method useful and easy to use.
- The connection between SPI efforts and improved value production together with the challenges found in core capabilities became better visible to the participants (especially to higher management) in a single measurable and traceable way. The method showed that they had quite differing perspectives among them on stakeholder values and initiatives which could have not been found in their "normal" setting of SPI related investment decisions.
- The moderated process was helpful both for collecting a broad range of candidates for benefits and change initiatives and for effectively reducing (selection and prioritization) the high number to the most important candidates.
- The process needs a competent moderator to guide the group and avoid wasting energy on less interesting issues. Additionally, selection of motivated group members is also important.
- Tool support was found to be useful in effectively collecting and analysing knowledge. The tool takes between 120 and 180 minutes to execute the method in one workshop. It was acknowledged that without tool support, the proposed method could take too long to be reliable. It may take five to six hours to execute the method without the provided tool support. It would certainly be very difficult to sort, filter and aggregate results manually. More importantly, participants felt that *it was very exciting and motivating to see the results on screen with such tool.*
- Tool support greatly reduced the effort and time taken to execute the method and helped focus on the brainstorming and consensus building.
- The strength of linking between change initiatives, company capabilities, and stakeholder values could easily be captured in the process. As an example, assume two of the stakeholder values as fast "product development time" and "on time delivery". One could argue that formalizing current software process may help to achieve them, or maybe an agile SPI initiative like Scrum may be used. However, through the alignment in iteration I of the method, we saw that the capability of "defect prevention" had the strongest impact on fulfilling "fast product development time" and "on-time delivery". Subsequently, we found which SPI initiative was best to improve defect prevention capability? The iteration II showed It was not Scrum but Extreme programming. Such linking of different issues would be difficult without systematic method, the proposed could be one.

4.3 Limitations

One could argue that the success of the proposed method depends on the contributions of the method users, which may induce personal bias voting and building consensus. One way to address this concern is to have a highly competent moderator. Another limitation of the method is in tool support which in current prototype stage only provides the matrices and scoring but does not help in discovering emergent patterns from the alignment data in the matrices. This feature and the tool support's web-based application are under development. Web-based individual control of votes may also help reduce the possible bias in the execution of the method.

5 Conclusions and Future work

This article[5] reported the key experiences on applying a customized method to better align stakeholder value propositions with SPCs and SPI Initiatives. The experience from an active research cooperation

[5] The longer version of this paper with detailed worked out example is available online at www.vaspo.org/alignmenttool

showed that software companies need a systematic approach for their informed decision on identifying SPI initiatives related investments. Furthermore, software companies also need to identify investment targets that are likely to generate substantial increase in stakeholder value.

The experience with software process assessment models were found useful to give an overview on relevant SPCs, but also showed major risks of a mismatch between process areas and companies' needs for improvement of SPCs. These experiences formed motivation to develop a method that helps reduce the risk of misalignment between SPCs and SPI. Additionally, we have been witnessing increasing interest from software companies to apply this method for better understanding their company specific alignment issues. We aim to conduct more empirical workshops and report more detailed publication when the themes emerging from the empirical analysis saturates. This will also contribute to research issues in SPI based on grounded empirical results. Our experience from the application of the method also indicates that higher level of abstraction in 'SPI alignment with Stakeholder values' will contribute to risk management body knowledge in software engineering

As a result, in the article, we presented a QFD-based method that supports the alignment of stakeholder benefits with SPCs and SPI initiatives. Furthermore, we reported on experiences from applying the method and relevant tool support in workshops in the VASPO SPI research project and with a Finnish software company. The proposed work is getting positive response from the Finnish software industry. At least two software companies have already requested to execute the proposed method in their setting. Subsequently, future work for this research is to conduct multiple empirical studies and find out the practical impact of the method after its implementation in the software companies. As a result, a research framework will then be developed, fully grounded on empirical validation, to realize both practical and research impact from this ongoing work.

Acknowledgments: We sincerely want to thank Dr. Jyrki Kontio, the founder and a former professor of the Software Business Lab, for motivating the idea of method development, and the support from our industry partner in this work, Basware Oyj, Finland. We also like to thank all participants in the workshops for their participation, sharing their views and providing valuable feedback.

References

1. Boehm, B.: Developing Groupware for Requirements Negotiation: Lessons Learned. Software, IEEE, **18** (2001) 46-55
2. Boehm, B.: Value-Based Software Engineering: Reinventing "Earned-Value" Monitoring and Control",. SIGSOFT Softw. Eng. Notes, **28** (2003) 3
3. Dyba, T.: An Empirical Investigation of the Key Factors of Success in Software Process Improvement. Empirical Software Engineering, **31** (May 2005) 410-139
4. Erdogmus, H.: Return on Investment. Software, IEEE, **21** (2004) 18-22
5. Fayad, M. E., & Laitnen, M.: Process Assessment Considered Wasteful. Commun ACM, **40** (1997) 125-128
6. Govers, C. P. M.: What and how about Quality Function Deployment (QFD). International Journal of production economics, **46-47** (1996) 575-584
7. Liu, X. , Sun, Y., Kane, G. et al.: QFD Application in Software Process Management and Improvement Based on CMM. (2005) 1-6
8. Long, Carl., Vickers-Koch, Mary: Using Core Capabilities to Create Competitive Advantage. Organizational Dynamics, **24** (Summer 95) 6-22
9. López-Cortijo, R., Guzmán, J. G., Amescua Seco, A.: ICharts: Charts for Software Process Improvement Value Management. Software process improvement. 4th European Conference, EuroSPI 2007, Potsdam, Germany, September 26-28, 2007. Proceedings, (2007) 124-135
10. Prahalad, C.K., Hamel, G.: The Core Competence of the Corporation. Harvard Business Review, (May-June 1990)
11. Richardson, I., Murphy, E., Ryan, K.: Development of Generic Quality Function Deployment Matrix. Quality management journal, **9** (April 2002) 25-43
12. van Solingen, R.: Measuring the ROI of Software Process Improvement. Software, IEEE, **21** (2004) 32-38

Author CVs

Dr. Nilay Oza is a senior researcher at Software Business Lab and VTT Technical Research Centre of Finland with research interests in software business, value perspectives, SPI and qualitative research.

Prof. Stefan Biffl is an associate professor of software engineering at the Vienna University of Technology with research interests in Software Product and Process Improvement.

Mr. Christian Fruhwirth is a PhD student at Software Business Lab, Helsinki Univ. of Technology, Finland with research interests in software business models, SPI and risk management.

Ms. Yana Selioukova is a PhD student at Software Business Lab, Helsinki Univ. of Technology, Finland with research interests in SPI and requirements engineering.

Mr. Riku Sarapisto is a senior product development manager at Basware Oyj a software product company in Finland with research interests in practitioner values from SPI initiatives.

Appendix 1: Example of method execution (Tool support)

The following example is based on the experience in a workshop with the SPI specialists and a Finnish software company. The example can be mapped to method description presented in section 3. Figure 2 shows Iteration 1 which includes 4 steps. Circled numbers in Figure 2 and 3 indicate corresponding steps. The moderator keeps the control of the tool and moderates the workshop.

Iteration I: Aligning stakeholder values with SPC

(Elicit stakeholder values) shows that participants identified different stakeholder values from On-time delivery, Operating costs to Quality of support. Based on the voting the tool identifies top 10 stakeholder values and then based on moderated group negotiation, prioritized list of values are identified indicating which values are the most important (for example On-time delivery got highest importance in this example (Figure 2 shows importance in grey vertical bar)

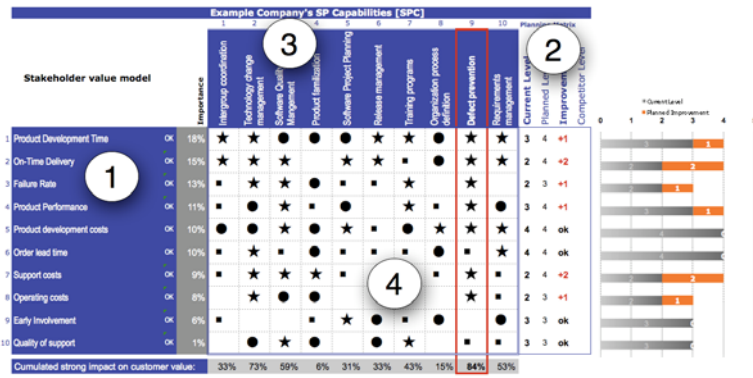


Figure 2: Iteration 1

(Organizational planning) shows that participants rate company's performance in delivering stakeholder values. Figure 2 shows that participants rate their current level of performance and planned (targeted) level for each stakeholder value. The tool also gives an option to rate performance

against the competitor or known benchmark (in Figure 2, we have not used any example).

(Elicit company's SPCs) shows that company's SPCs are identified based on the available list in the tool and added company specific SPCs. The identification and prioritization followed same as in step 1.

(Align SPCs with stakeholder values) shows that stakeholder values are aligned with the SPC using one of three available impact symbols. (Where no symbol was used the SPC had less then minor impact on the value). The impact is decided based on moderated discussion. In the presented example the capability of "defect prevention" is identified as having the strongest impact on fulfilling the most important stakeholder values, with its cumulated strong impact being the highest. These SPCs (referred to as "chosen SPCs" in the method) are prioritised based on cumulative impact and is used in next iteration for aligning them with SPI Initiatives.

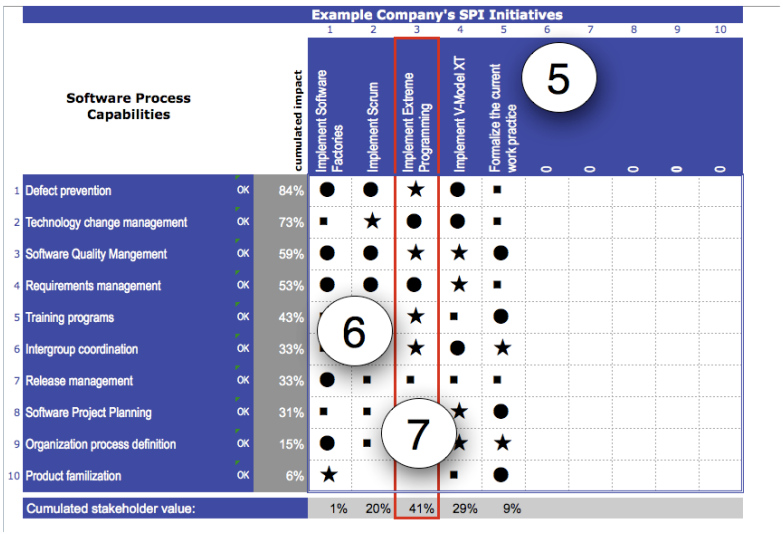


Figure 3: Iteration 2

Iteration II: Aligning SPCs with SPI initiatives

(Elicit SPI initiatives) shows that like in step 1 and 3, SPI initiatives are identified and prioritized. Figure 3 shows examples of five selected initiatives.

(Align SPI initiatives with chosen SPCs) shows that similar to step 4, the alignment is performed using impact symbols.

(Choose the SPI Initiatives which have the strongest impact on improving chosen SPCs) shows that similar to step 4, the cumulative impact of SPI Initiative on stakeholder value is counted for each SPI initiative against SPCs. For example, in Figure 3, implementing extreme programming shows highest possible impact with 41%. This concludes the method and further pattern analysis from symbols can be done. This helps decision makers to identify relevant SPCs and their alignment to SPI Initiatives.

Product Line Requirements Engineering in the Context of Process Aspects in Organizations with Various Domains

Alexander Poth

Abstract

Software product lines are used today in many domains. The obvious conclusion is that product lines are a useful concept. This paper describes practical experiences gained with the adaptation of product line processes and methods in projects in the automotive, finance, and telecommunication domain. It discusses generic topics and also gives a brief introduction into selected specific topics. The focus is on requirements management and engineering for product lines. The observations are consolidated in a reference model for consulting services in the product line context.

Keywords

SPI (Software Process Improvement), SPICE (ISO 15504), Requirements Engineering, Product Lines, Life Cycle, Development Model, Automotive, Finance, Telecommunication

1 Purpose and structure of the document

The purpose of this document is to explain how a method and process reference model is evolved from consulting activities in various projects and domains. The model's purpose is to improve the consulting services for process improvement in the requirements engineering domain with focus on product lines through reuse of this model or some of its aspects.

The model described herein is not an assessment model for ISO IS 15504 or an extension to CMMI like the family maturity framework (FMF). This reference model is intended as a guideline for practitioners during software process improvement.

Results of this paper are experience-based and driven by observations in different domains. These observations are consolidated in a universal approach. The consolidated approach can be tailored for specific organizations. This consolidated approach results in an experience-based model. The model covers aspects for processes as well as methods for systematic product line engineering. Continuous improvement of this model is guaranteed through integration of the lessons learnt in the projects.

2 Characteristics of product lines in various domains and organizational structures

The domains observed here have some commonalities through their use of product lines. But these commonalities are not necessarily the main motivators for the introduction or the maintenance of the product lines. In the following, some aspects of the present situation in the domains are described. These aspects have a direct or indirect influence on the discussions in the following chapters.

Automotive

The automotive industry strives to produce more vehicle lines based on a common electric/electronic architecture as the platform for the product lines. The main motivation for platforms is to take

advantage of economies of scale.

The automotive domain uses embedded software which is optimized for minimal resource consumption. This optimization does not allow high overhead for product line aspects.

The automotive domain achieved the customization of its products through the intensive use of parameters. Thus there are for the same functionality many sets of parameter values for product customizing.

If a product line is used in the automotive industry it is used over more vehicle lines [He03], [Bu04]. But these lines are not developed in parallel. This asynchronous development leads, for example, to changes to the product line through improvements of the functionality for later vehicle series. These changes to the functionality could occur over the entire production time of the first vehicle series of a product line.

Finance

The finance domain is interested in consolidating its IT to reduce costs. The results can be observed in the consolidation of data centers beyond the borders of banks. These inter-banking IT service providers are interested in a universal product line which is customized (for example by using parameters and “add-on”-functionality) for the customer bank. The core of these systems should be as similar as possible for all customers in order to optimize development and maintenance costs.

An important aspect of these product lines is that these systems are in operation for a long time and must be maintained over this time. Keep in mind that even today many COBOL applications are still in operation.

Processes and use cases of the applications or products are grouped according to functionality. The collaboration of all these groups is the IT system of the bank. This collaboration is often orientated toward services. The product line or service architecture is constantly changed by modifications which are motivated by new laws or new application functions demanded by the bank.

Telecommunication

The telecommunication domain can be divided into two groups: the classical telecommunication component provider, like infrastructure components providers, and the telecommunication service providers, such as internet service providers.

The component providers are affected by a high frequency of new product development cycles. To realize this high frequency they have to reuse parts of other products. The reuse automatically implies product lines. The result is that component providers [Bo05] are mainly driven by the same aspects as the automotive domain for their product lines.

The service providers’ main objective is to offer services for the market. The services are directed, for example, to different customer groups such as private and business customers. The services are based however on the same product line which offers the different characteristics demanded by customers. The result is that service providers are primarily driven by the same aspects as the finance domain for their product lines.

The frequency of product renovation or update is higher in the telecommunication domain than in the automotive and finance domain.

Common aspects of all domains

An orientation toward software development standards such as SPICE (ISO IS 15504) or CMMI can be observed in all domains. In some domains, these standards are tailored, e.g. automotive SPICE.

Furthermore, all domains strive for a shorter time-to-market. The telecommunication domain has a shorter product development time in comparison to the automotive domain. A reduction of development time however causes the same problems in all domains with respect to optimal use of resources, efficient product line management, process optimization for faster cycle time etc., because the domains are operated at their specific limits.

All domains use projects to realize their undertakings. For example, the project life cycle focuses on development aspects and not on the complete product life cycle. Furthermore, the entire product life cycle is shorter than the product line life cycle, if it is assumed that the product line is the platform for more products which are not totally synchronized on the market. The potential conflict here is that the objective of the undertaking is different from the project or product line objective. In an example from a finance project, the development project responsible wishes to reduce development costs by reducing the maintainability of the product line source code. Furthermore, there is often a lack of awareness of resource usage or architectural considerations by other undertakings which interfere with each other. This was extensively observed in a telecommunication organization. An institution external to the project or undertaking is needed to manage these conflicts.

Product documentation is a relevant topic for all domains because all must make extensions or changes to the products in the future. This implies that all domains need to know what is inside a product release – so they need a relation between releases and requirements.

With respect to the size of an organization, it is noticeable that in midrange-sized companies the administrative overhead for controlling product complexity is the same as in large companies. The activities for handling complexity are always the same and could not be simplified on the process level. The difference is that in the midrange-sized company many roles are handled by one person. In all domains and company sizes there is a lack of human resources at key points in the development life cycle because they are costs and should be minimized. For example, software testing is not an element in the value creation chain (or added value) from a development view point. The same product development life cycle implicates that all companies have the same requirements life cycle with the only differences being in the number and/or semantics of a requirement life cycle state.

An interesting aspect of the service idea is that a service component itself is part of a product line. The service component is highly adaptable to realize a service component network which delivers the service or product. In this context the definition of the terms service, product line and function is highly dependent on the specific wording of the organization units of the company like Marketing [He07] or Development. Furthermore, check if the context is not a set of „product populations“ [Gr04] which make the borders even more unclear. Keep the wording aspect in mind while interpreting this reference model.

3 The requirements life cycle in a product line context

Requirement life cycles are defined for example in RAM [Go07], but there the context is neither the product line context directly, nor does this model cover the entire life cycle up to the product retirement. Furthermore, this model is not compatible with SPICE for different reasons, e.g. because RAM does not identify system requirements.

Based on these motivations this chapter briefly describes a product line requirement life cycle in the context of SPICE or CMMI. On this abstraction level the requirement life cycle is the same in all domains, but in different companies it can vary with regard to specific characteristics.

The basis for systematic requirements engineering is structured requirements elicitation. Elicitation is “asking” the customer what he wants. Systematic elicitation is not only a use-case description but also the focus on non-functional requirements (NFR) [Do05]. The NFR are important in the product line context because aspects such as reliability, safety, performance etc. can have a noticeable impact on the entire product line.

After elicitation, the requirements need a structure. A good practice is feature-based, for example, [He03] or, more precisely, function-orientated. This is the main criterion for organizing requirements which have the NFR as cross-cutting [Be01] aspects.

The analysis of the requirements can be driven with focus on the undertaking. The target design is set on top of the product line architecture or other reference architecture. In the next step the undertaking defines the changes or extensions to the architecture. For a better integration it is useful to show what happens in other undertakings because this can influence the future target architecture as well. This is the latest point for a synchronization of the different undertakings to reduce their competing influence

on the architecture.

The product line must select some of the requested functions with their architectural modifications for implementation into the next releases. The selection is necessary because undertakings have more requests than can be implemented in one step or development iteration.

The undertaking implements the requirements iteratively and incrementally in the releases of the product line. The undertaking must check whether the requirements are realized in a functionally correct manner. Furthermore, the responsibility for the fulfillment of the customer requirements lies with the undertaking. The responsibility for the fulfillment of technical and operational requirements lies with the product line. This is a strict separation of the customer and technical responsibilities.

The product documentation can be set up by the consequential forwarding of requirements from the undertakings to the release which implements the requirement. This systematic propagation documents for example the present situation of earned value of the undertaking and on a release view the functionality in each release.

4 Structured integration into a systematic product line engineering

In the next chapters the domain context and observations are integrated into a process and method model. This methods and process pattern could be tailored for companies and their organizations.

The constraint for the development of the model was not to have negative influence on the conditions of the domains (which are described above) and thus lead to a high acceptance by the users.

Wording for the context of the model presented

- An undertaking is container for a customer demand. This demand typically is realized by a project or change request.
- The product line is the system and a product is an instance of the product line. This definition is necessary because the product line itself has to be logically consistent. If the product line is consistent an instance of it is also consistent.

To make the system operational it needs an architecture which defines some constraints for the entire product line or architectural platform. If a function requires some change for its integration into the system, coordination between different undertakings becomes necessary. These undertaking requirements could be in conflict with the requirements of other undertakings or with the product line strategy.

4.1 Classification scheme and states for requirements

Requirements are classified by a defined scheme of types. With this scheme it is possible through annotation to create relations between the customer requirements and the responsible on the realization side. In general, a customer requirement could be a requirement of the type “what” or “how”. The “how” is a pre-defined solution which is expected by the customer. As a pre-defined solution is a constraint for all following development activities it should be avoided. Nevertheless, customer requirements can be defined on each abstraction level, see figure 1.

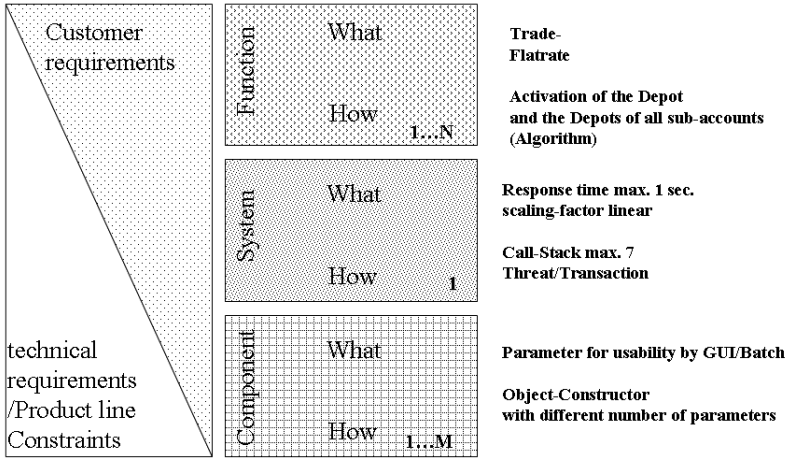


Figure 1: Abstraction levels for requirements and their origin.

In comparison to RAM this new approach distinguishes between customer and realization side requirements. As a result, in this model requirements are divided up into three requirement types: requirements on the function, the system and the component level. The requirements of all three levels can be of the type "What" or "How" as described earlier. That scheme is shown in the right part of figure 1. Additionally, there is a second dimension to distinguish requirements (left side of figure 1), that is the classification in requirements from customer or realization-side. The detailed determination of the classification of the requirements into function, system, or component has to be defined individually for the domain and organization. From an abstract point of view, the functional level is a description of the functionality of the product, the system requirements are the refinement of the function in a realization (design) context. The component level is the specification of the details of the realization and based on the system requirements. These are the three basic requirement levels which have to be used for a systematic development based on standards like SPICE. If more levels are needed by a specific undertaking, those can be defined.

For identification and subsequent grouping of functions use cases are useful. Typically, use cases are documented by means of UML and additional textual explanations. For preparation, templates for the structure of the documentation of the functions have proven to be a useful tool. For the systematic elicitation of the related non-functional requirements it is recommended to use the method [Do05].

Product lines realize complex systems which are realized by components. A component for example in the finance domain is the account management application which groups all accounting functions. The accounting functions nowadays are often realized as small services inside account management. In the automotive domain the components are ECUs (electronic control unit) which implement the functions.

As a conclusion at this point it is important that all requirements are related to an abstraction level. For complex systems it could be necessary to add more abstraction levels. Furthermore, a component in complex IT systems could be for example a package.

Also of importance is that it is clear what are input requirements and what is the solution of the abstraction levels because the solution on a higher level is the input for lower levels. And if it is clear what a solution is and what not it is easier to change a solution transparently to another if technology shifts in the future. The observation here is that many functions exist longer than their realizations in a special technology and this justifies a good functional design which is independent of the realization.

With this classification it is possible to track the progress of the requirements into the product line. An example state model for this tracking is shown in figure 2.

Functional-"What"→Functional-"How" →
System-"What"→System-"How"→
Component-"What"→Component-"Who"→
Implemented →
Released →
Retired

Each LC-Phase is traceable by a Intra-state:

Requirements elicitated →
Ready for review →
released

Figure 2: Generic requirements life cycle

4.2 A Generic Variability Model

Examples for modeling variability can be found in [Bu04] or [He03]. The variability model is an important part of product line engineering but in this context a generic model is useful which is described in this chapter.

The benefit of the model is that it is possible to document variability bottom-up and or top-down. Bottom-up is the engineering view and top down the marketing view of variability. The objective of variability modeling is to document the variability on the abstraction level on which the variability initially occurs under the aspects which cause the variability. The variability is like a grouping level for the individual contexts. This implies that variability exists on the function, system and component level. In figure 3 an example is shown for the functional level which is explained here. The variability in the function level only is modeling the functional variability. The functional variability is separated in physical and logical variability. The functional variability on the function level is for example the difference between the functionality based on variant points. For the automotive wiper example it is a wiper with and without interval wiping. Between functions it is possible to model the dependencies. These dependencies could be constraints like exclude or include for cases which are not complex. For complex dependencies rules are useful. Physical grouping is in the functional context a grouping of functionality which fits together. This grouping often is a pre-selection on the functional level for small functional systems which fit and work together. This physical grouping could be used for the derivation of systems. The logical grouping is for example the marketing aspect for defining product bundles.

If more abstraction is needed each grouping level could be stacked by modeling for example different aspects of physical grouping like body types or markets (these are not motivated by functional aspects like homologation).

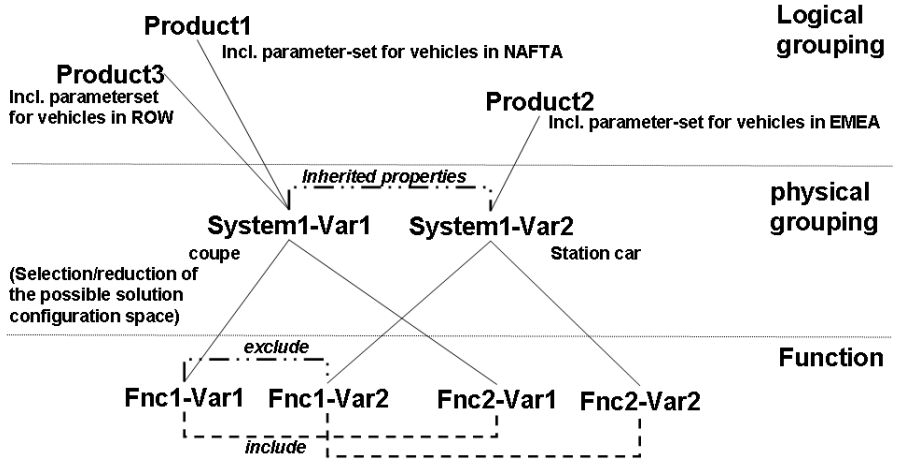


Figure 3: Functional variability and the respective configurations up to the product level

4.3 Organizational views of the product line

Now that a basis for systematic requirements engineering is defined for our context, the next step is to extend this basis to include organizational aspects. As described before the development of undertakings is implemented in projects. As a result of these projects a product will be produced or changed. Then, the products are sold in releases. One can regard these changes of the product releases as the "variability in time". The various products of one product line can be seen as the "variability in space". Different products can differ either in parameters only or they can be different executable code binaries. Figure 4 shows the three organizational responsibilities - the undertaking- (or enterprise-), the product- and the version-responsibility - for realizing a product from the first definition of the requirements to a final instance of the product line.

The three dimensions correspond to typical roles in an organization: the project manager, the product manager and the release manager. Each role has its individual view. The (development) project manager watches only his (development) aspects of the undertaking. The (technical) product manager is interested in long term goals as well as in the strategies behind the product line. The release manager focuses on operational aspects such as stability and performance of the release only. There are other roles that influence the product line, e.g. the portfolio manager, the operation staff or the sales responsible with guarantee aspects; but although these roles are important in real projects, they do not have to be considered in this overview.

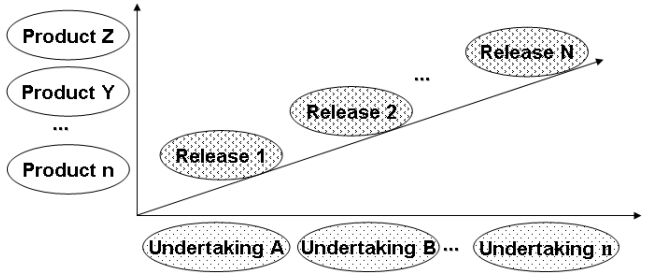


Figure 4: The 3 dimensions: undertaking, product and release

Efficient collaboration requires established processes for product lines. However, process models like SPICE or CMMI do not have this in their scope. So the implemented processes have to cover the requirements of the process model and the product line model without exerting any negative impact on each other. For example, the process for the product line which is used by the undertaking must not have a negative impact on the project view in a SPICE assessment.

The process can be seen as a set of activities around a pool of requirements or functions of different undertakings. This pool is filled up with pre-analyzed and prioritized functions by the undertakings. From this pool, the product line management selects functions for the next release. The selected functions have to fit into the human resources planning of the organization.

Figure 5 shows the process in a descriptive manner. The activity "undertaking requirements analyses" analyze the requirements, taking into consideration the wishes of the customer, the "customer requirements", and the "current state" of the product. The undertakings transfer the target state – the pre-vision of the final "undertaking target state" in the figure - and the pre-analyzed and prioritized requirements to the product line analysis (filling up the requirements pool) – the "product work orders" in figure 5. The product line activity "product requirements analyses" defines an overall undertaking target state – the "product target state" in the figure. All undertakings in the portfolio can be synchronized no later than here. The last step is to define the content of functionality for the next release through the selection of the prioritized requirements with a focus on feasibility.

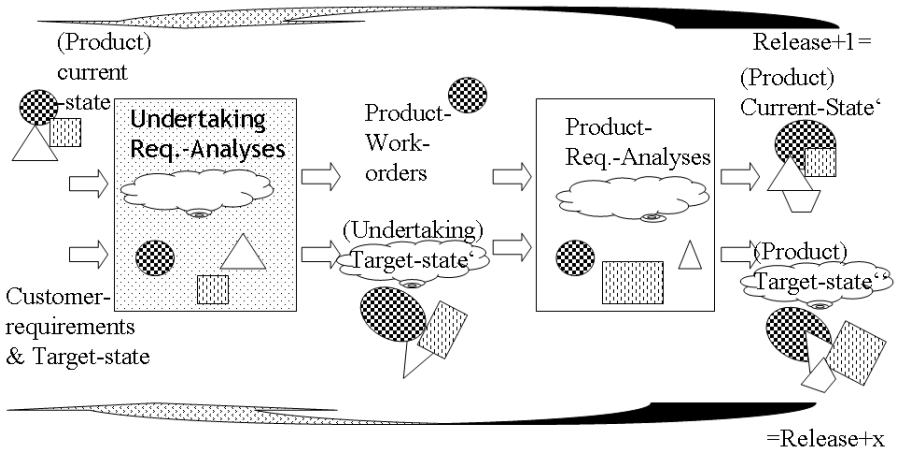


Figure 5: Decoupling of undertakings and product via releases

Undertakings record requirements from the view of users and application experts. They perform the analysis and prioritization from the perspective of the customer or the undertaking. Furthermore, they create a high-level design for the respective undertaking. The undertaking monitors the implementation of the requirements and the „moving target". If it detects a deviation from the target state, it changes or re-prioritizes the requirements (or work orders) for the next iteration or increment of the implementation.

The product management must guarantee that the requirements of the undertakings conform to the strategy of the product line architecture. This implies a release-specific (fine-)design across all undertakings and a release-based documentation.

For better synchronization between undertaking, product and release, it is useful to couple the asynchronous processes through quality gates (figure 6). The upper process is for the iterative and incremental undertaking process. The lower is the sequence of the release implementation process which delivers products as undertaking outcomes of a release.
These quality gates can be defined "unclearly" at the end of the corresponding phases to make the process more conform to a RUP development process. Nevertheless, the synchronization elements have to be fulfilled.

This view is not a direct representation of the reference framework for product management [Br06]. First of all it is focused on requirements engineering and second of all it is orientated toward projects for realizing undertakings. This is not a conflict, however, but only a question of perspective.

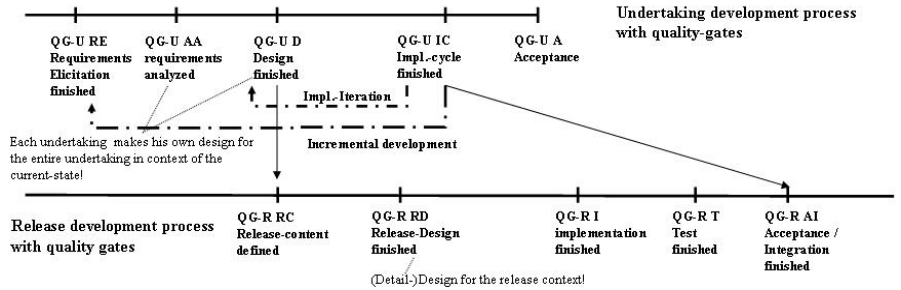


Figure 6: Undertaking development process and the release product process

5 Conclusion/lessons learnt from the use in the domains

This integrated reference model is used for the gap analysis in software process improvement projects within a product line context. An improvement potential is identified on the basis of the gaps between the model and the company. Most of the companies with an established process model have this model at least partially in use. Based on the gap between the established processes and the model, it is easier to argue for process or method improvements if the consultant gives recommendations based on experiences which are running in other organizations. Furthermore, it provides an integrated "vision" to the argumentation. Later on the SPI the model is useful to argue that everything in the model is state of the art or based on observations and experiences of other companies.

Lessons learnt from the projects are also integrated into the reference model for continuous improvement of the consulting quality.

The difference to the reference models like [Br06], [Go07] etc. is that these models concentrate on special aspects of the life cycle. The purpose of the model is to remedy the lack of an integrated life cycle. The model is an inspiration for improvements all around the requirements life cycle within a product line context.

This overview of the reference model does not address specialized topics like variability or complexity management. These special parts of experience-based details of the model are the large part of real improvement projects but the problem at the detail level is the reusability in other organizations.

In future investigations and improvements of the model we have to optimize the component-based product line aspects in order to provide better support for service-oriented product lines.

6 Literature

[Be01] Bergamans, L., Aksit M.: Composing Crosscutting Concerns using Compositions Filters. Communications of the ACM Oct 2001, Vol. 44, No. 10 51-57

[Bo05] Bosch, J., Software Product families in Nokia. International Software Product Line Conference (SPLC) 2005, 2-6

[Br06] Brinkkemper, S., Bijlsma, L., Nieuwenhuis, R., Versendaal, J., Weerd, I.: A Reference Framework for Software Product Management, ISSN: 0924-3275

[Bu04] Bühne, S., Lauenroth, K., Pohl, K., Weber, M.: Modeling Features for Multi-Criteria Product-Lines in Automotive Industry. Workshop on Software Engineering for Automotive Systems (SEAS), at ICSE 2004

[Do05] Doerr, J., Kerkow, D., Koenig, T., Olsson, T., Suzuki, T.: Non-Functional Requirements in Industry - Three Case Studies Adopting an Experience-based NFR Method. Proceedings 13th IEEE International Conference on Requirements Engineering (2005) 373 – 384

[Go07] Gorschek, P., Garre P., Larsson, S., Wohlin C.: Industry Evaluation of the Requirements Abstraction Model. Requirements Engineering Journal, Vol. 12, No. 3, pp. 163-190, 2007

[Gr04] Groningen, R.: Building Product Populations with Software Components. PhD-Thesis Universität Groningen 2004

[He03] Heumesser, H., Houdek. F.: Towards Systematic Recycling of System Requirements. International Conference on Software Engineering 2003 (ICSE'03), 512-519.

[He07] Helferich, A., Schmid, K., Herzwurm, G.: Softwareproduktlinien für Anwendungssysteme: eine Analyse aus Techniksicht und Marktsicht. Das Wirtschaftsstudium 5/07, Düsseldorf, S. 686-692

7 Author CV

Alexander Poth

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

Since 2003 he has been working on requirements engineering methods and concepts. In 2004 he qualified as an iNTACS SPICE assessor, in 2006 as an iSQI Professional for Project Management and in 2007 as an ISTQB Test Manager and iREB Professional for Requirements Engineering.

Since 2007 he is working as senior consultant for the SQS AG with focus on quality management in the automotive, telecommunication and finance domain. Since 2004 he has been working as a quality management consult and engineer. Previously he worked as a software developer from 2000 for a telecommunications OEM and an ERP System OEM.

MPS.BR: A Successful Program for Software Process Improvement in Brazil

Mariano Angel Montoni, Ana Regina Rocha, Kival Chaves Weber

Abstract

Software Process Improvement (SPI) implementation based on software process reference models and standards is a complex and long-term endeavour that requires investment of large sums of money. These obstacles usually hinder organizations from implementing SPI successfully, especially for Small and Medium-size Enterprises (SME) that operate under strict financial resources. This paper describes the MPS.BR, a nationwide program for SPI in Brazilian organizations. The main goal of this initiative is to develop and disseminate a Brazilian software process model (named MPS Model) aiming to establish a feasible pathway for organizations to achieve benefits from implementing SPI at reasonable costs, especially SMEs. This paper presents the main components of MPS Model and discusses the strategy executed to establish and maintain a community of MPS Model practitioners. The results of MPS Model adoption and dissemination in Brazilian software industry are also presented in this paper.

Keywords

Software process reference model; software process assessment model; ISO/IEC 15504; SPICE; CMMI; lean implementation of SPI.

1 Introduction

Software Process Improvement (SPI) implementation based on software process reference models and standards is a complex and long-term endeavor that requires investment of large sums of money [1]. These obstacles usually hinder organizations from improving software processes, especially for Small and Medium-size Enterprises (SMEs) that operate under strict financial constraints. For instance, software process reference models have been adopted by very few Brazilian organizations. Until 2003, in India, 32 organizations were assessed as CMM Level 5; while in China there were one organization assessed as CMM Level 5 and in Brazil none (in total, only 30 Brazilian organizations had gone through CMM based assessments until 2003) [2]. This scenario was considered critical, especially for SMEs that usually struggle to implement SPI. Considering that Brazilian SMEs (fewer than 50 employees in small organizations and between 51 and 100 people in medium-size enterprises) employ more than 182 thousand people that corresponds to 56% of the total people employed by Brazilian software organizations [3], there was an urgent need in Brazil to increase software development capabilities, especially on SMEs.

This paper describes an initiative to improve software processes in Brazilian organizations named MPS.BR Program. MPS.BR is the acronym for the Portuguese expression *Melhoria de Processo do Software Brasileiro* (Brazilian Software Process Improvement). This initiative started in 2003 under the coordination of Association for Promoting the Brazilian Software Excellence (SOFTEX), and is a joined effort of Brazilian government, software industry and research institutions. The main goal of this initiative is to develop and disseminate a Brazilian software process model (named MPS Model) aiming to

establish a feasible pathway for organizations to achieve benefits from implementing SPI, especially SMEs. The model was developed based on international standards and internationally recognized models and best practices for SPI implementation and assessment, and also on Brazilian software industry business needs.

The next section discusses some background in the SPI area. Section 3 presents the organizational structure of MPS.BR Program necessary to develop the MPS Model. Section 4 presents the MPS Model main components. The main results regarding adoption and dissemination of the MPS Model in the Brazilian software industry are presented in section 5. Finally, section 6 presents the conclusions.

2 Background

The ISO/IEC 15504 standard provides a comprehensive and generic approach for the model-based assessment of process capability [4]. ISO/IEC 15504 defines requirements for performing process assessment that fall into two classes: requirements associated with the performance of the assessment itself, and requirements concerned with the process model that forms the basis of the assessment [5]. The processes that are defined as a comparison target for a particular assessment are defined in a Process Reference Model (PRM). ISO/IEC 15504-2 specifies the contents and basic structure of PRMs. Each process in a PRM is described in terms of its purpose which are the essential measurable objectives of a process, and outcomes of its implementation [4][5].

In order to support process definition and process assessment and improvement, the ISO/IEC also initiated an effort to develop a PRM within the domain of software engineering. The base standard for such initiative was the ISO/IEC 12207 [6]. This standard provides a comprehensive group of life cycle processes, activities and tasks for software products and services. The ISO/IEC 12207 was extensively revised and these revisions were published in the form of two amendments that provide process purpose and outcomes to establish a PRM in accordance with the requirements of ISO/IEC 15504-2. The PRM defined in ISO/IEC 12207 Amendments 1&2, associated with the process attributes defined in ISO/IEC 15504-2, establish a Process Assessment Model (PAM) used as a common basis for performing assessments of software engineering process capability [5].

An important internationally recognized model for software process improvement is the Capability Maturity Model Integration (CMMI) [7]. CMMI is a process improvement maturity model for development of software products and services developed by the Software Engineering Institute (SEI). Nevertheless, recent studies have been executed aiming to analyse the reasons why software organizations avoid adopting CMMI. For instance, Staples et al. [8] shows that software organizations, especially small ones, will never gain any benefit from process capability maturity improvement because they consider it infeasible to adopt CMMI mainly due to the small organization size, the high costs involved in providing SPI services, and the lack of time to dedicate on SPI activities. Coleman and O'Connor [9] also present a study of how SPI is applied in the practice of software development. Their study results show that many software managers reject to implement SPI models because of the associated implementation and maintenance costs.

Considering the great difficulties associated to the implementation of SPI initiatives, many empirical studies conducted investigations about the critical factors that influence the success of SPI initiatives. For instance, Wangenhein et al. [10] and Cater-Steel et al. [11] point out that the main issue is to convince SMEs on the expected business benefits, and recognizes the need to minimize the costs for process assessment and to make the benefits of SPI initiatives visible in a short time frame.

In this context, some SPI initiatives worldwide are starting to be developed focusing on SMEs. The ISO set up a working group on SE Life-Cycle Profiles for Very Small Enterprises (VSEs; companies with fewer than 25 employees) [12]. The Software Engineering Institute initiated a CMMI in Small Settings project to provide approaches, tools, techniques, and guidance in small settings (defined in this project as organizations or companies of fewer than approximately 100 people and projects of fewer than approximately 20 people) [13]. National initiatives are also starting to flourish as increases the needs for more adequate models to deal with characteristics of specific markets. For instance, the Mexican Ministry of the Economy developed a standard [14] based on ISO/IEC 12207, including practices from ISO 9000:2000, CMMI, PMBOK, SWEBOK, and ISO/IEC 15504. Moreover, in Ireland dur-

ing 2006, Enterprise Ireland (the development agency with responsibility for small companies) initiated a three-year plan for developing and promoting software quality through improved processes [15].

3 MPS.BR Program: Developing a Brazilian Model for Software Process Improvement

The main problems that inhibit organizations from adopting software process reference models, such as CMMI, ISO/IEC 12207 and ISO/IEC 15504, as reported in the SPI literature, are related to SPI implementation, maintenance and assessment costs, and difficulty to convince organizations of the benefits of SPI. In this context, the Association for Promoting the Brazilian Software Excellence (SOFTEX) decided to start a nationwide initiative, named MPS.BR Program, for improving software processes of Brazilian organizations. Therefore, the main goal of the MPS.BR Program was to develop and disseminate a Brazilian software process model (named MPS Model) aiming to establish a feasible SPI implementation and assessment pathway for organizations to thrive, especially SMEs.

A special concern during the conception of the MPS.BR Program was not only to develop a model aligned to Brazilian software industry business needs, but also to establish and maintain a strategy to guarantee adequate adoption and dissemination of the model throughout the whole country. In order to achieve this goal, the MPS.BR Program considered the following requirements: (i) the MPS Model should incorporate internationally recognized best practices for software process implementation and assessment, and also consider Brazilian software industry business needs; (ii) the costs for implementing and assessing software process should be reasonable to SMEs that operate under strict resources; (iii) a community of MPS Model practitioners should be established and maintained to provide adequate MPS Model based implementation and assessment services; and (iv) the MPS Model should be adopted by a large number of Brazilian software organizations, especially SMEs.

The MPS.BR Program has been executed since 2003; it is coordinated by the SOFTEX - a private not-for-profit organization created to promote Brazilian software industry competitiveness - and it is sponsored by the Brazilian Ministry of Science and Technology (MCT), the Brazilian Research and Projects Financing Agency (FINEP), the Brazilian Service of Support for Micro and Small Enterprises (SEBRAE), and the Inter-American Development Bank (IDB), but it is being increasingly sustained by revenues from MPS services.

In order to manage the MPS.BR Program, an organizational structure was defined. The MPS.BR Program Structure units are the following: (i) MPS.BR Program Team: responsible to manage the program activities; (ii) MPS Technical Model Team: responsible to develop and maintain the model, and to prepare and execute MPS model trainings; and (iii) MPS Accreditation Forum: responsible to certify organizations to provide MPS model-based implementation and assessments services, to evaluate and control implementations and assessments results, and to ensure that organizations certified on the MPS model execute their activities within expected ethical and quality limits.

4 MPS Model

The ISO/IEC 12207 and ISO/IEC 15504 were used as the technical base elements for defining the MPS Model Components. Considering the importance of CMMI model for Brazilian organizations that operate in international markets, the MPS Technical Model Team also considered the CMMI as a complementary technical base element for the MPS Model processes definition. The MPS Model is constituted of three main components: the MPS Reference Model; the MPS Assessment Method; and the MPS Business Model. Next, we describe in details these components.

4.1 MPS Reference Model

The MPS Reference Model (MR-MPS) is documented in the form of three guides: the MPS General

Guide, the MPS Acquisition Guide and the MPS Implementation Guide. The MPS General Guide provides a general definition of the MPS Model and common definitions to all other guides. The MR-MPS is a 'candidate conformant' to ISO/IEC 15504 since it fulfils the requirements for a PRM defined in ISO/IEC 15504-2. The MR-MPS processes are described in terms of their specific purpose and outcomes used to evaluate specific process implementation. Each process defined within the MR-MPS has unique process descriptions and identification and the set of process outcomes are necessary and sufficient to achieve the purpose of the process. The MR-MPS processes are an adaptation of the ISO/IEC 12207 Amd 1 & Amd 2 processes and the CMMI-DEV process areas. The MPS General Guide also provides a definition of scope and composition of MR-MPS process profiles for a declared level of organizational maturity level. A maturity level consists of process outcomes and process attributes achievement results for a predefined set of processes. Therefore, the MR-MPS maturity levels are defined in two dimensions: process capabilities dimension and process dimension.

The MR-MPS process capabilities dimension is constituted of a measurement framework for the assessment of process capability based on the processes defined in the MR-MPS processes dimension. Process capability is defined on an ordinal scale that represents increasing capability of the implemented process, from not achieving the process purpose through to meeting current and projected business goals. Within this measurement framework, the measure of capability is based upon a set of process attributes (PA). Each attribute defines a particular aspect of process capability. The MR-MPS process attributes are based on the ISO/IEC 15504-2 process attributes used to define capability levels. The MR-MPS defines nine PA: PA 1.1 (process performance attribute); PA 2.1 (performance management attribute); PA 2.2 (work product management attribute); PA 3.1 (process definition attribute); PA 3.2 (process deployment attribute); PA 4.1 (process measurement attribute); PA 4.2 (process control attribute); PA 5.1 (process innovation attribute); and PA 5.2 (process optimization attribute). Each PA comprises a set of Process Attribute achievement Result (PAR) used to evaluate a specific PA implementation.

The MR-MPS process dimension is constituted of the processes to be assessed. The MR-MPS process dimension describes seven sequential and accumulative groups of processes that correspond to the MR-MPS maturity levels. The seven MR-MPS maturity levels are: A (Optimizing), B (Quantitatively Managed), C (Defined), D (Largely Defined), E (Partially Defined), F (Managed) and G (Partially Managed). The level G is the most immature level and level A is the most mature one. The MR-MPS maturity levels (ML) processes profiles were defined accordingly to specific business needs of Brazilian software industry. The most relevant need was to make the benefits of SPI initiatives visible in a short time frame at reasonable implementation and assessment costs.

A process shall be assessed up to and including the highest maturity level defined in the assessment scope. The combination of process outcomes and attributes achievement results and a defined grouping of processes together determine the organizational maturity level. Table 1 presents MR-MPS processes and the PA that shall be added to each maturity level.

Table 1: MR-MPS maturity levels (ML), processes and process attributes (PA)

ML	Processes	PA
A	Causal Analysis and Resolution	1.1, 2.1, 2.2, 3.1, 3.2, 4.1*, 4.2*, 5.1*, 5.2*
B	Project Management (new outcomes)	1.1, 2.1, 2.2, 3.1, 3.2, 4.1*, 4.2*
C	Decision Analysis and Resolution, Risk Management and Development for Reuse	1.1, 2.1, 2.2, 3.1, 3.2
D	Requirements Development, Product Design and Construction, Product Integration, Verification and Validation	1.1, 2.1, 2.2, 3.1, 3.2
E	Human Resources Management, Process Establishment, Process Assessment and Improvement, Project Management (new outcomes) and Reuse Management	1.1, 2.1, 2.2, 3.1, 3.2
F	Measurement, Configuration Management, Acquisition and Quality Assurance	1.1, 2.1, 2.2
G	Requirements Management and Project Management	1.1, 2.1

** This PAs are applicable only to selected processes. All the other PAs must be applied to all processes.*

A correspondence can be delineated between MR-MPS and CMMI maturity levels. The processes profiles of MR-MPS maturity levels F, C, B and A correspond respectively to the processes profile of CMMI maturity levels 2, 3, 4 and 5. The processes profile of MR-MPS ML G corresponds to an intermediary level between the processes profile of CMMI maturity levels 1 and 2. The processes profile of MR-MPS maturity levels E and D are two intermediary levels between the processes profile of CMMI maturity levels 2 and 3. The MR-MPS organizes the processes profile differently than the CMMI for two reasons: (i) to provide a more feasible pathway for capability maturity growth by reducing the number of processes to be implemented in the first (and riskier) maturity levels, and (ii) to facilitate the visibility of SPI investments results in a short time of frame. These characteristics make the MPS Model more appealing to SMEs that operate under strict resources and need to prove by themselves the benefits from implementing SPI. Moreover, the MPS.BR is conformant with ISO/IEC 15504, since it satisfies the requirements for a software process reference and assessment model.

Besides the MPS General Guide, the MPS Model contains other 2 guides. The MPS Acquisition Guide describes an acquisition process for software and related services, and its purpose and outcomes are conformant with the ISO/IEC 12207 international standard. The MPS Acquisition Guide also identifies recommended practices for software acquisition such as in IEEE STD 1062 [16]. The MPS Implementation Guide provides technical guidance for implementing the seven MR-MPS levels. The MPS Implementation Guide describes theoretic concepts that fundaments the processes defined in the MR-MPS maturity levels. Moreover, it contains detailed information on implementing MR-MPS process outcomes. This guide establishes the means to uniform Software Engineering knowledge among SPI practitioners and to reduce the risk of misunderstanding implementation issues essential for successfully satisfying MR-MPS processes outcomes.

4.2 MPS Assessment Method

The purpose of process assessment is to determine the extent to which the software processes contribute to achievement of organizational business goals and to help it focus on the need for continuous software process improvement. According to ISO/IEC 15504-2, an assessment should be carried out against a defined assessment input utilizing conformant Process Assessment Model(s) related to one or more conformant or compliant Process Reference Model(s).

In order to satisfy ISO/IEC 15504-2 requirements for a Process Assessment Model, the MPS Technical Model Team defined the MPS Assessment Method (MA-MPS) and documented it in the form of the MPS Assessment Guide. This guide also describes the assessment process defined to support the application of the MA-MPS. The MPS Assessment Guide also defines the requirements for accreditation of: Organizations to provide MPS assessments services (MPS Assessment Institutions); MPS Competent Assessors; and MPS Provisional Assessors (assessors that support competent assessors during assessments). The objective of the assessment method MA-MPS described in the MPS Assessment Guide is to verify the maturity of an organization unit in the execution of its software processes.

One important aspect to the success of the MPS Model is that the assessment method should be executed aiming to minimize assessment costs without compromising the reliability and consistency of the assessment results. Therefore, the assessment method activities were defined aiming to optimized the required effort and, as consequences, making assessment costs not prohibited to software organizations, especially SMEs. For instance, the amount of effort necessary to assess an organization in the MR-MPS maturity level G is only 40 man-hours (including the initial and the final assessment). The complete assessment of MR-MPS maturity level A processes in a specific organization would require approximately 200 man-hours. The low effort for assessing MR-MPS processes is obtaining mainly due to the characteristics of the MPS Assessment Method. For instance, in the final assessment, the assessment team only revisit the processes outcomes that had problems identified in the initial assessment. This practice helps to focus the attention in the final assessment on the problems in the processes implementation.

4.3 MPS Business Model

In order to guarantee the success of the MPS Model, it is essential that organizations can effectively adopt it and achieve benefits from implementing SPI. Therefore a component, named MPS Business Model (MN-MPS), was developed aiming to support MPS Model adoption and dissemination.

The MN-MPS comprises two types of SPI implementation models according to organizations specific needs and availability of resources: (i) a Specific Business Model suitable to large companies which do not want to share MPS Model based SPI services and costs with other companies; and (ii) a Cooperative Business Model for groups of SME interested in implementing and assessing the MPS Model, and sharing MPS services and costs, for instance, training activities. The Cooperative Business Model is especially attractive to SMEs, because the implementation based on this model is feasible of obtaining external funds to support SPI implementation and assessment activities.

5 Adoption and Dissemination of the MPS Model in the Brazilian Software Industry

One criterion for evaluating the success of standards and models is to identify the degree of acceptance and usage by their target community [17]. In this context, a specific goal was established in the context of the MPS.BR Program addressing MPS Model adoption and dissemination across the country. This goal is two fold: (i) to capacitate MPS Model based consultants and accredit institutions to provide MPS Model based implementation and assessment services in different cities of the country; and (ii) to support adoption of MPS Model by a large number of organizations, especially SMEs.

In order to capacitate MPS Model based consultants and accredit institutions to provide MPS Model based services, a strategy, constituted of official courses and exams, was defined and is being executed to establish and maintain a community of MPS Model practitioners. So far, more than 2,800 people attended MPS Model courses in different cities of Brazil, and more than 900 people were successfully approved in MPS Model exams.

Another important aspect to guarantee the adoption of the MPS Model is to accredit institutions in different parts of the country to provide MPS Model based services. 18 organizations were accredited to provide MPS Model-based implementation services and other 7 organizations were accredited to provide MPS Model-based assessment services in different regions of Brazil. More than one hundred MPS Model implementation consultants are effectively working in MPS Model based initiatives in Brazilian organizations. Moreover, in total, 38 competent and provisional assessors are associated to accredited MPS Assessment Institutions with conditions to conduct MPS Model based assessments across the country.

In order to support adoption of MPS Model by a large number of organizations, the SOFTEX (coordinator of the MPS.BR Program) organizes groups of organizations according to the MPS Cooperative Business Model for SMEs interested in implementing and assessing the MPS Model, and sharing MPS services and costs. Sponsors such as IDB, MCT and SEBRAE provided 40% to 50% of the overall MPS Model based implementation and assessment costs of those groups. Moreover, these organizations can also share other costs, for instance, training activities. By integrating those groups, organizations can significantly reduce financial resources necessary to improve their processes. The implementation of MPS Model according to MPS Cooperative Business Model has been recognized by SMEs as an important pathway to achieve process improvements benefits at reasonable costs.

Figure 1 presents the number of organizations assessed in the MPS Model. Until May 2008, 100 organizations had gone through successful MPS Model based assessments. We can observe in figure 1 that the majority of MPS Model assessments is in the lowest MR-MPS maturity level G (69% of the total assessments). This high number shows that MPS Model is attractive to organizations seeking process improvement, but that do not have sufficient resources to commit to initiate large improvement cycles. 80% of the organizations that were assessed on the two bottom maturity levels (G and F) are SMEs that implemented the MPS Model under the Cooperative Business Model. All the assessment results are published on the SOFTEX Web site (www.softex.br/mpsbr).

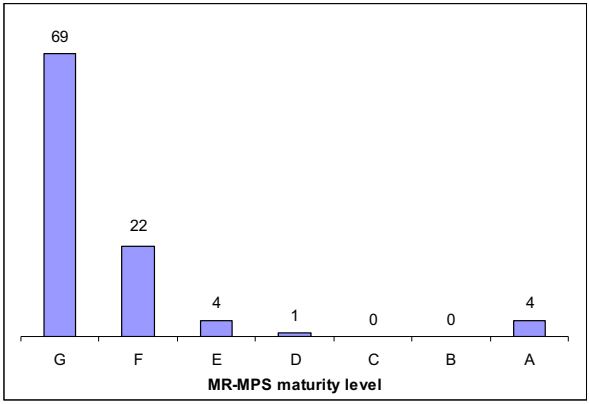


Figure 1: Number of organizations assessed in the MPS Model.

6 Conclusions

This paper presented the basic structure of the MPS.BR Program and the main components of the MPS Model, a software process model developed to address Brazilian software industry needs.

We conclude from the results of MPS Model adoption and dissemination that it facilitates SPI in Brazilian organizations. This conclusion is supported by two observations identified in MPS Model based SPI initiatives. Firstly, the seven maturity levels of the MPS Model help to diminish SPI complexity and to increase SPI initiatives control by reducing the number of processes to be implemented at each maturity level and by facilitating the achievement of SPI benefits visibility in a shorter term. Secondly, the MPS Business Model and the strategy to promote the community of MPS Model practitioners help to diminish inherent risks of SPI initiatives, such as, lack of financial and specialized human resources.

7 Literature

[1] Goldenson, D.R., Gibson, D.L. Demonstrating the Impact and Benefits of CMMI: An Update and Preliminary Results, SEI Special Report, CMU/SEI-2003-SR-009, Oct. (2003)

[2] Veloso, F., Botelho, A. J. J., Tschang, T., Amsden, A. Slicing the knowledge-based economy in Brazil, China and India: a tale of 3 software industries. Report. Massachussetts Institute of Technology (MIT), Sep. (2003)

[3] SIBSS-SOFTEX, 2008. IBGE. Special tables, 2005 – to be published (2008)

[4] Rout, T.P., El Emam, K., Fusani, M., Goldenson, D., Jung, H.-W.: SPICE in retrospect: Developing a standard for process assessment. Journal of Systems and Software (2007), doi: 10.1016/ j.jss.2007.01.045

[5] ISO/IEC 15504. Information Technology–Process Assessment. Part 1 – Concepts and vocabulary; part 2 – Performing an assessment; part 3 – Guidance on performing an assessment; part 4 – Guidance on use for process improvement and process capability de-termination; and part 5 – An exemplar proc. ass. model.

[6] ISO/IEC 12207:1995/Amd 1:2002/Amd 2:2004. Information Technology – Software Life Cycle Processes.

[7] SEI: CMMI® for Development (CMMI-DEV), V1.2. Software Engineering Institute (2006)

[8] Staples, M., Niazi, M., Jeffery, R., Abrahams, A., Byatt, P., Murphy, R.: An exploratory study of why organizations do not adopt CMMI. Journal of Systems and Software 80 (2007) 883-895

[9] Coleman, G., O'Connor, R.: Software process in practice: A Grounded Theory of the Irish software industry. Vol. 4257 NCS. Springer Verlag, Heidelberg, D-69121, Germany, Joensuu, Finland (2006) 28-39

[10] Wangenheim, C.G.v., Varkoi, T., Salviano, C.F.: Standard based software process assessments in small companies. Software Process: Improvement and Practice, Vol. 11 (2006) 329-335

[11] Cater-Steel, A., Toleman, M., Rout, T.: Process improvement for small firms: An evaluation of the RAPID assessment-based method. Information and Software Technology 48 (2006) 323-334

[12] ISO/IEC NP TR 29110, Software engineering -- Lifecycle profile for Very Small enterprise (VSE), http://www.iso.org/iso/iso_technical_committee.html?commid=45086

[13] CMMI in Small Settings Toolkit Repository, Software Eng. Inst., 2004; www.sei.cmu.edu/cmmi/ acss.

[14] Information Technology—Software—Models of Processes and Assessment for Software Development and Maintenance, tech. report NMX-059-NYCE-2005, Ministry of the Economy, Mexico, 2005.

[15] Richardson, I., Richardson, I., Gresse von Wangenheim, C.: Guest Editors' Introduction: Why are Small Software Organizations Different? Guest Editors' Introduction: Why are Small Software Organizations Different? Software, IEEE 24 (2007) 18-22

[16] IEEE STD 1062:1998. IEEE Software Engineering Standards Collection. IEEE Recommended Practice for Software Acquisition, IEEE STD 1062 Edition. New York NY (1998).

[17] Meek, B., 1996, “Too soon, too late, too narrow, too wide, too shallow, too deep”, StandardView 4 (2), pp. 114-118.

8 Author CVs

Mariano Angel Montoni

MPS.BR Authorized Implementation Practitioner and Competent Assessor, Software Engineering Consultant, Researcher and D.Sc. student at the Federal University of Rio de Janeiro. He received his B.Sc. in Computer Science from UFBA, Salvador, Brazil, and his M.Sc. in Software and Systems Engineering from COPPE/UFRJ, Rio de Janeiro, Brazil. Contact him at COPPE/UFRJ - Federal University of Rio de Janeiro, POBOX 68511 – ZIP 21945-970 – Rio de Janeiro, Brazil; mmontoni@cos.ufrj.br.

Ana Regina Rocha

MPS.BR Model Team Technical Coordinator, MPS.BR Authorized Implementation Practitioner and Competent Assessor, Software Engineering Consultant, Researcher and Professor at the Federal University of Rio de Janeiro. She received her B.Sc. in Mathematics from UFRJ, Rio de Janeiro, Brazil, her M.Sc and D.Sc. in Software and Systems Engineering from PUC, Rio de Janeiro, Brazil. She is a member of the Federal University of Rio de Janeiro. Contact her at COPPE/UFRJ - Federal University of Rio de Janeiro, POBOX 68511 – ZIP 21945-970 – Rio de Janeiro, Brazil; darocha@cos.ufrj.br.

Kival Chaves Weber

MPS.BR Program executive manager, Brazilian Software Quality and Productivity Program (PBQP Software) general manager and Consultant for Quality, Productivity, Innovation and Competitiveness in the software industry. He was Secretary of Informatics in the Brazilian Ministry of Science and Technology (MCT) and Principal of Brazilian Software Companies. He is co-author of 2 books and over 80 articles on Software Quality and Productivity. He received his B.Sc. in Telecommunication Engineering from IME - Military Institute of Engineering, Rio de Janeiro, Brazil, and his M.Sc. in Electrical Engineering from COPPE/UFRJ, Rio de Janeiro, Brazil. Contact him at SOFTEX - Association for Promoting the Brazilian Software Excellence, POBOX 6123 – ZIP 13081-970 - Campinas - SP – Brazil; kival.weber@nac.softex.br.

The Marriage of two Process Worlds

Bernhard Sechser, Continental AG

Abstract

In this paper the experiences are described in the approach to combine two different process worlds from two different companies into one. It wants to give you an understanding that several factors are influencing this approach, like cultural differences, different languages, process design methods or the process architecture. They all have an effect on the definition of a global process definition. You will also see that it is important to define tailoring conditions so that the result is flexible enough to be used on any project.

Keywords

Process definition

Process architecture

Tailoring

Pilot projects

Process elements

1 Motivation

The life of a process engineer is never boring, even if he is working in a company that grows continuously. Beside of the always running improvement initiatives the organizational situation has a not insignificant influence to the construction of processes.

Last year two big companies in the automotive sector – Continental Automotive Systems and Siemens VDO – merged into a new big company with a lot of synergies and improvement opportunities. To use this opportunities as fast as possible the management forced the definition and integration of common processes and IT applications. Within half a year a common engineering process including system, software, electronics and mechanics disciplines and all supporting and management processes like project, configuration or quality management should be established. A great and nearly unreachable goal if you think of the totally different histories of the two companies.

But the process engineers on both sides have obtained a lot of experience in the last years in merging processes, so the starting point was a good one.

Figure 1: The motto of the new company

2 How to come together – The differences in culture

Doing a closer look to these two worlds you will see that in reality not only two processes must be combined. During time the different organizational areas in each of the two companies have "improved" their specific processes in such a way, that the common basis was sometimes not visible any more. Therefore the risk was high that the number of people involved in the integration meetings grows up to an amount, where discussions never seem to come to an end instead of agreeing compromises. So the number of participants must be decreased to the relevant stakeholders. But who are these stakeholders?

To identify these persons is difficult even when the new organizational structure is unclear short time after the merge. But on both sides so called "Expert Groups" exist, who already worked on a common approach before. Out of these groups the experts for the new integration teams are nominated, assuming that they can speak for the whole group.

It seems that everything is prepared now to start working. What is the working material the integration team has to deal with? You have the process content, described in different details, and maintained in different process tools. The tool … seems that this is not a topic for a process integration team, but far wrong. The tool influenced in a strong way the kind of modelling processes because different methods and techniques are supported, and therefore also the way to come to a common description was influenced.

Another difference was the way how process management was handled in the two companies. On one side a big organization with process executives and process owners were defined taking care, that everything was documented and detailed in a solid way. On the other side the process experts were nominated out of the development team, describing the necessary content with enough flexibility to adapt it to the project's needs. Both approaches have their advantages and disadvantages, but it is not easy to commit to common detailing level with the uncertainness how the future organization will look like.

3 The child is born – A common language is necessary

A child is able to learn any language. Even if the parents are speaking different ones, it can assign the different words to one of its parents. But what if the parents do not understand each other?

So first they have to establish a common glossary of terms where they describe the meanings of all expressions both parties are using. And hopefully they can agree in each case to a common expression. For the further activities it will be much easier then to discuss and decide about details of the new process.

Don't think that this topic is an easy one. A lot of time in such integration meetings is spent with discussions about simple names like "Software Designer":

> "We do not have Software Designers; we only have Software Engineers that are also doing the software design!"

Is this really a problem? It shouldn't be! In most cases it is an organizational problem. To understand that a common process on an abstract level cannot reflect all organizational roles is an essential part to understand. Such details can be brought into the process later by adapting it to the business units needs.

4 Planning the children's room – The process architecture

Often it happens that the child is in the world but nobody took care about the children's room. How shall it look like? How shall it be structured? This does not mean to have a detailed plan which toy comes on which place, but to define the "places" where the toys can be put on before buying them.

In our process language we will talk about the process elements of which the process shall consist, and the associations between them. In the beginning it should not be too detailed but the main elements should be mentioned:

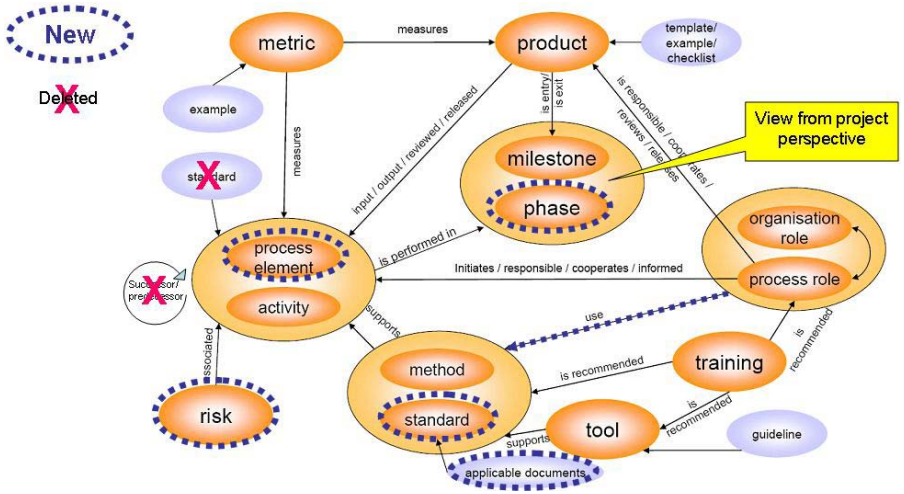


Figure 2: Working draft of a common meta model

- Phases and milestones: The frame for a process landscape can be described with a series of phases, finalized by a milestone or gate. In each of these phases a set of activities are performed

that not necessarily belong to one process area, e.g. “Planning software tests” can be in the same phase as “Designing the software architecture”.

- Process elements and activities: A group of single activities that belong together in any way can be described as process elements, e.g. “Identify Software Requirements” as part of the process element “Software Requirements Analysis”. Each activity should be assigned to a phase where it will be performed.
- Products: The result of each activity should be reflected in an outcome. This needs not to be a document, can also be a database entry, a tool setup, a binary file, some kind of hardware part, … These outcomes can also be used as input for another activity. In the end you might perhaps be able to establish an activity-product flow.
- Roles: Someone needs to perform all the defined activities. Therefore roles are necessary. They are associated to the activities as well as to the products. But take care that no inconsistencies occur in the triangle Role <-> Activity <-> Product <-> Role. With the right tool you are able to avoid these inconsistencies automatically.

With these four elements a first version of a common process description should be possible.

5 Building the children’s toys – Defining the processes

Now that we know how the children’s room shall look like we can start to design the toys. Which kind of toys do we need? What is the area where the child wants to play? Does it only want to play with “Software Engineering” bricks, or also with “Project Management” trains, or even with “Customer Management” towers?

We have to take care for all process areas, but we need to define priorities and responsibilities.

In most cases we have different process owners and groups for the different areas, so they can work in parallel. But they have to harmonize their interfaces, not to detect in the end that the necessary input for one area is never created from anyone.

Which areas are the most important ones? Those ones that must be used in the next future. This is difficult to decide and depends on the way of future collaboration. Leaving this answer open we will concentrate us on the software engineering process.

It is useful to define a structured approach for working on the different process elements. One example could be the following:

- Phase names and descriptions per process
- Milestone names and descriptions per process
- Deliverables per milestone (part of output products)
- Deliverable descriptions
- Role names and descriptions per process
- Activity names and descriptions per phase
- Input/output product names and descriptions
- Association of activities, input/output products and roles

Keeping this order in mind it should be possible to put more and more details into the process without running the risk of discussing details that are not necessary at this time. In the end we should have a first version of a process description that can be used and adapted by each project type or business area.

6 "Cloning" the toys – How to adapt the process to Individual needs

Now we have defined the ideal children's room – for the ideal child. But does this ideal child exist? No! Each child is different; each one has its own needs. So the next challenge will be to use the predefined toy room and modify it in a way that our child can use it and is happy with, but without modifying it in a way that the original purpose is not transparent any more.

To reach this it is necessary to define "tailoring conditions", which can be used on the original process to derive the specific process. One condition should always be that each element can be detailed in a necessary way without deleting the mandatory elements from the original process (see figure 3).

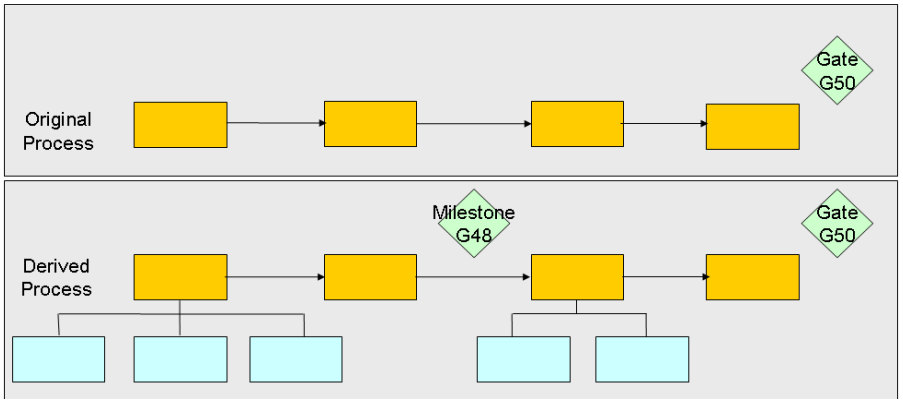


Figure 3: Detailing of process elements

Another possibility is to define "optional" elements in the original process. These elements need not to be used in a derived process and can be replaced by individual ones (see figure 4).

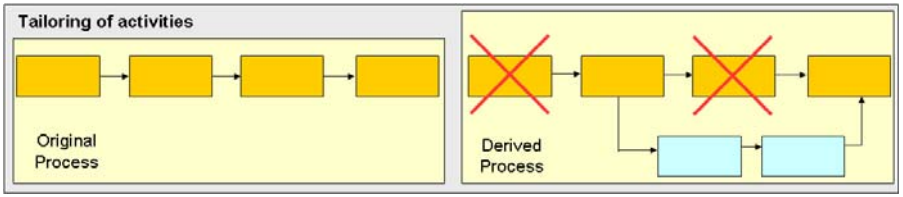


Figure 4: Deleting of optional elements and defining of additional elements

In the end we should have a process definition that fulfils the needs of the project and the demands from the original process.

7 The child is playing – The processes are living

And now child … close your eyes … open the door to the child's room … open your eyes and … let's play! Let's see if the toys we designed for our child are really making fun, or if it is unhappy with one or all of the toys.

The setup of pilot projects is a useful way to find out, if the processes defined on the "green table" are

really able to work with it. In fact the process was not only defined by theoretical experts but by practitioners, so the risk of having completely wrong processes is very low. But details in the handling of different process areas might not be defined well enough, or a template does not match with the demands of an activity, or a role description misses an essential part …

The opportunities for improving the processes are big … as it was before … before the merge. But the processes live, and that was the goal.

8 Conclusion

You have seen that combining two process worlds into one is not an easy task. You have to take care of different situations like cultural background, different languages, process modelling tools, individual needs and last but not least the users of the process. To do this in such a short time frame is possible if you focus on the main parts and not lose in details. And there will be time to add as many loops of process improvement as you need after you have established a first version. Because the best child is not good enough if it was never born.

Author CVs

Bernhard Sechser

Bernhard Sechser studied computer science in Erlangen at Friedrich Alexander University. He started his business career as a software developer at Siemens AG and later at Conti Temic microelectronic GmbH in Nuremberg, now part of the global Continental Corporation. He used his experiences to establish a process oriented software development. After his certification as Bootstrap- and SPICE-Assessor (2001) he participated in numerous assessments. As certified iNTACS Competent Assessor he is member of ASQF e.V. and a founding member of iNTACS e.V. Today he leads the Quality Management department for System Quality Assurance in-Business Unit Transmission at Continenta

A Methodological Framework for Reengineering Improvement

Marcos Ruano-Mayoral, Inmaculada Puebla-Sánchez,
Ricardo Colomo-Palacios & Ángel García-Crespo

Abstract

The present paper presents a Framework which is adaptable, in the first place, to the requirements of organizations, and in the second place, to the characteristics of the products which are subject to the reengineering process. The focus of improvement of the reengineering process is realized in the model by means of continuous inspection and supply. The application of the framework in the core of the project in which it was developed has achieved, on the one hand, a successful outcome by the development team of the project and the organization in which it was implemented, and on the other hand, the fulfilment of the objectives of reengineering.

Keywords

Software Reengineering, Migration Plans, Reengineering framework

1 Introduction

Many organizations dedicated to the development of software packages face the problem of having to migrate or reengineer their products, adapting them to new technologies and functionalities. In the case of software manufacturers, the requirements of modifying their technology imply the modification of the client base, significantly multiplying the effects.

The present initiative stems from the collaboration of Universidad Carlos III de Madrid in a project committed to the reengineering of a tool developed by a Spanish company. The main objective of this work is to combine and normalize the amount of tasks emerged from the reengineering requirements in software-intensive organizations. The proposed framework represents a reference for the documentation elaboration and the evolutionary normalization of the tasks within an environment that allows the assessment and the continuous improvement of the involved processes, with a view to the generalization of practices for reengineering other products developed by the company.

The remaining sections of the paper have been structured as follows. Section 2 presents the state of the art, in relation to Software Reengineering. Section 3 describes the Reengineering Framework developed, and lastly, Section 4 outlines the main conclusions derived from the work.

2 State of the art

The concept of Reengineering was described by Hammer and Champi (1993) as the fundamental rethinking and redesign of business processes to archive dramatic improvements in measures of performance. The application of the concept of reengineering in the software field began to gain importance towards the end of the 1980s (Arnold, 1992), alongside reference to past development, efforts in order to reduce maintenance expense and improve software flexibility (Premerlani & Blaha, 1994). Further simpler and more general definitions of software reengineering also emerged in the literature. For example, experts have stated that reengineering may be viewed as any activity that either improves the understanding of a software or else improves the software itself (Arnold, 1993). Bearing in mind the latter definition, the activities which comprise the process of software reengineering may be divided into two types (Kullbach & Winter, 1999). The first kind of activities are concerned with understanding, such as source code retrieval, browsing, or measuring. The second kind of activities aims at evolutionary aspects like redocumentation, restructuring and remodularization.

The role of software reengineering is vital, as it decreases complexity, increases quality and better equips future business environments (Behling, Behling & Sousa, 1996). It has been considered of such importance that since its initial application in productive environments, it has been regarded as an engineering problem (Feiler, 1993) that includes most aspects of traditional software development with additional constraints. As a consequence, since the middle of the 1990s many initiatives have emerged for planification (Sneed, 1995), the establishment of a life cycle (Manzella, Mutafelija, 1992), taxonomies (Chikofsky & Cross, 1990) and economic aspects (Sneed, 1991). It is evident that it was not until the millennium decade that efforts were realised to place reengineering in an organisational environment, taking into account all the types of constraints which may be faced by a process by such characteristics. In the Software Engineering Institute of the Carnegie Mellon University, Bergey, O'Brien & Smith (2001) identified a model for Software Migration Planning, which has been applied to software implemented in the American Department of Defense. The framework is based on the management of the migration effort as the critical factor.

Taking as a basis the characteristics of the model developed by Bergey, O'Brien & Smith (2001), the project team of Universidad Carlos III de Madrid has constructed a framework which adapts itself to the characteristics of reengineering of commercial software packets, and additionally, is a continuous improvement framework for the reengineering processes of an organization.

3 Reengineering framework

According to the specific requirements stated before, this article presents a Reengineering Framework built over two complementary methodologies closely related to the construction of information systems and software process. Firstly, ESA (European Space Agency) (ESA, 1991) software development standards PSS-05-0 has been adapted. These development standards will provide the guidelines for the code generation processes and will be the basis for the definition of the tasks related to the generation and analysis of the different solutions for a specific migration project. Secondly, we base our framework in Software Engineering Institute's Software Migration Plan proposed by Bergey, O'Brien & Smith (2001). This recommendation defines a set of phases and tasks suitable for every reengineering process, and represents a relevant reference for the development of this kind of projects due to its origin and contents. The migration guidelines presented in the recommendation are complemented by a guide of migration practices extracted from the work by Bergey, Smith & Weiderman (1999) and developed at SEI too.

After performing a process of analysis and definition of the solution, a framework for the reengineering process developed has been established. Figure 1 depicts the structure of this process.

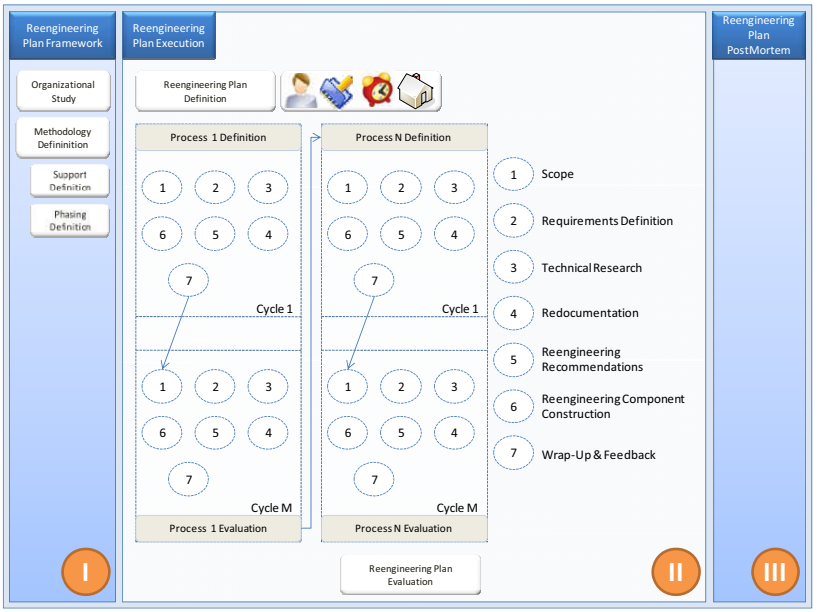


Figure 1: Reengineering Framework.

The proposed Framework is fundamentally based on two premises. The first one is the need of a framework for the improvement of reengineering processes not only in the particular migration project in which it is deployed but in the whole of the organization. The second one is the will of obtaining a flexible framework, without a stiff structure that hinders its adaptation to the specific scenario of the organizations where it is deployed.

I. Reengineering Plan Framework. It is a stage prior to the Reengineering Plan that is aimed at the design of the framework that will guide the process as a whole. This stage is the aggregation of the following phases:

I.1. Organizational Study. An analysis of the different organizational elements involved in the reengineering process is performed. Some of the aspects covered by this analysis are the current circumstances of the organization and the target system, the market characteristics and the involvement of the different organizational levels.

I.2. Methodology Definition. The main outcome of this phase is the definition of the reengineering methodology that best fits to organizational needs having in mind the conclusions extracted from Organizational Study and the different methodologies available at the market.

I.2.1. Support Definition. The objective of this phase is to establish and define the support requirements for the execution of the methodology in the target organization. These requirements are considered to be essential to the reengineering process and are totally independent of the tasks and activities performed in each process and cycle.

I.2.2. Phasing Definition. A definition of the different phases that structure the methodology. Those phases are established at this point since they are the basis of the Reengineering Plan Execution.

II. Reengineering Plan Execution. During the execution stage the actual migration tasks are performed. The suggested Reengineering Plan Execution is based on the Reengineering Plan Framework applied in the VV context.The Reengineering Plan Execution stage defines a set of phases that are detailed next:

II.1. Reengineering Plan Definition. In this phase the characteristics of the migration to be tackled are defined. These characteristics include criteria relative to the number of reengineering process to be performed, the scope of each of the processes, resources allocation and task plan.

II.2. Reengineering Process Definition. Each of processes that make the plan up is an element that is tackled as an individual project for which scope and dependencies must be established. According to the Reengineering Plan Process, an arbitrary number of Reengineering Plan Processes can be instantiated to fulfil reengineering requirements in a phased approach. Each Reengineering Plan Process can be constructed by means of the following activities and tasks:

II.2.1. Process Plan. The Process Plan establishes the overall characteristics of the process including resources, plans, cycle descriptions and the products to be generated in each of the phases.

II.2.2. Process Cycle. A cycle is an activity that integrates a set of tasks aimed at the reengineering of a planned and specific element or set of elements. Each cycle comprises the execution of a combination of the following tasks:

II.2.2.1. Cycle Scope. This task specifies the cycle in a formal way, stating the cycle's objectives, a detailed plan of resources and time, the organization of the rest of the tasks of the cycle and an in depth definition of the products obtained from those tasks.

II.2.2.2. Requirements Definition. The Requirements Definition task is focused on the elicitation and codification of the reengineering requirements concerning the defined scope.

II.2.2.3. Technical Research. Technical Research is aimed at increasing the knowledge acquired by the development team about the possible solutions to be adopted and their implications. And adequate execution of this task will guarantee a collection of reengineering recommendations strongly based on the reality concerning the product to be reengineered.

II.2.2.4. Redocumentation. Provided that, after the Technical Research task, the existing documentation in the field of study is found to be poor or inadequate, it is planned to allocate the necessary resources for its re-ellaboration.

II.2.2.5. Reengineering Recommendations. From the requirements obtained in II.2.2.2 and the Technical Research the reengineering recommendations for each of those requirements will be specified.

II.2.2.6. Reengineering Component Construction. During this task the element established at the Cycle Scope is built. The development of this component will meet the previously elicited requirements and will apply the recommendations obtained in task II.2.2.5. The development process to be performed in this task will follow the guidelines of the PSS-05-0 standard for software development provided by ESA.

II.2.2.7. Wrap up & Feedback. This task sums up the whole cycle and documents it with the objective of being a feedback for the next cycles of the process.

The default organization of these tasks has a sequential nature and within a regular cycle all the tasks from II.2.2.1 to II.2.2.7 should be performed. However, it is possible to adapt this organization to the specific needs of a given cycle. This way, some tasks could be parallelized, i.e. II.2.2.3 and II.2.2.4, and even some of them could be discarded, but having in mind that tasks II.2.2.1 and II.2.2.7 are compulsory since they define the limits of each cycle.

II.2.3. Process Assessment. This task is aimed at the evaluation of the developed reengineering process, determining its completion and its contribution to new eventual reengineering processes.

II.3. Reengineering Plan Evaluation. Once that all the Reengineering Plan Processes have concluded, the quality of both the product and the reengineering process is evaluated. Additional evaluation is performed over the results of the reengineering, whose real process ends in this phase.

III. Reengineering Plan Postmortem. The aim of this stage is to provide as much feedback as possible to the reengineering framework to introduce improvements in both its definition and quality.

4 Case study

The VV organization (fictitious name) is a corporation founded in the early 1990s. VV has got a presence in 10 countries and customers in more than 80 countries. In 2007 earned a 3,7M€ turnover that represents a 60% increase. R&D is one of the pillars of VV, thus they invest a 20% of revenue in that area.

The company's most relevant product is ToM (fictitious name) with over 1200 licenses sold all around the world. ToM is a solution for administrative management of organizations that range from companies of all sizes to government organizations. The development of ToM involves almost 100 people from R&D, Technology and QA departments working for the inclusion of new functionalities, the adaptation to changing environments and the customization for specific clients of the solution out-of-the-box. The development process is supported by an in-house developed methodology inspired in the European Space Agency Standards for software development projects.

VV's objective is that ToM retains all of the functionalities developed for the tool, in its "Out of the Box" versions as well as in its customised versions developed by the clients. Without a doubt, the architecture of the tool demanded by the client should have the capability to be modified to adapt to new technologies for the creation of new user interfaces. Given this environment, the project accomplished by the collaboration of the university and the company has as objective the provision of a methodological framework, and continuous improvement of the reengineering process. This latter requirement stems from the necessity to implement the framework developed not only in the context of the ToM tool, but

also in the remaining product range of the company. On the other hand, it should be stated that one of the reasons for applying the reengineering process is because of the scarce documentation which accompanies the ToM tool, developed during the technology boom without the application of rigorous standards.

As an initial test of the preliminary acceptance of the proposal, a pilot project for the deployment of the framework was performed. The project team has divided into two groups with the same competential capabilities. Each of the groups was assigned a complete reengineering cycle with equivalent size and complexity. One of the groups adopted the proposed framework while the other followed the own internal methodology of the company. After finalizing the pilot project several structured interviews were conducted in order to inspect team members' opinion. The comments of the practitioners were categorical. The ones that did not use the new framework stated that they felt lost while performing their tasks since they were not aware of the real progress of each task and they did not have any way to clearly define the phases, the processes and the requirements of their project. This circumstance leaded them to accumulate a temporal deviation representing the 15% of the estimation. On the other hand, the work team that adopted the new framework only showed a 3% deviation. However, they praised the capacity of the framework to express the big picture of the problem and situate the practitioner on each task. Additionally they estimated as positive the need of performing post-mortem tasks after every phase, whether it is a cycle or a process.

After the pilot phase, the framework was applied to the reengineering of one of the core components of ToM. The project started with an initial process in which, after the inspection of 500,000 lines of code of the legacy components, 101 reengineering requirements where elicited in 350 versions. Those requirements where transformed in 102 reengineering recommendations with 241 versions. The reengineering recommendations were used to build 40 new classes that amount 8,000 lines of non-autogenerated lines of code.

5 Conclusions

From the point of view of the deployment of the proposed reengineering framework in a real production environment, two conclusions can be extracted. The first one is the improvement in the communication of the objectives of the reengineering process, its phases and products among the reengineering team. This conclusion stems from the qualitative feedback received from the resources involved in the process a stated in the previous section.

The second conclusion is the applicability of the solution to reengineering environments. The framework has proven to be flexible and adaptable to the environment where it was deployed permitting to meet the reengineering requirements.

As future works, two differentiated lines are proposed. Firstly, the quantitative experimentation of the developed framework by means of its deployment over two comparable projects. Such experimentation should tackle many different aspects ranging from product and process quality to the fulfilment of schedules and temporal and human resources consumption. Secondly, the extensibility of the framework should be studied aiming at different kinds of organizations and products.

6 Literature

Arnold, R. S. Software reengineering: a quick history, Datamation, 15 May 1992, pp. 13-14.

Arnold. R. S. A Roadmap Guide to Software Reengineering Technology. In Software Reengineering. IEEE Computer Society Press, 1993.

Behling, R., Behling, C. &Sousa, K. Software Re-engineering: Concepts and Methodology in Industrial Management & Data Syst. Vol. 96, No. 6, pp. 3–10. 1996.

Bergey, J.; O'Brien, L. & Smith, D. (2002). DoD Software Migration Planning. Technical Note CMU/SEI-2001-TN-012. Disponible en http://www.sei.cmu.edu/pub/documents/01.reports/pdf/01tn012.pdf

Bergey, J.; Smith, D. & Weiderman, N. (1999). DoD Legacy System Migration Guidelines. Technical Report CMU/SEI-99-TN-013. Disponible en http://www.sei.cmu.edu/pub/documents/99.reports/pdf/99tn013.pdf

Chikofsky, E.J. & Cros, J.H. Reverse Engineering and Design Recovery: A Taxonomy. IEEE Software, Volume 7 , Issue 1 (January 1990), pp: 13 – 17.

European Space Agency (1991). ESA Software Engineering Standards ESA PSS-05-0. Issue 2. Disponible en ftp://ftp.estec.esa.nl/pub/wm/wme/bssc/PSS050.pdf

Feiler, P. Reengineering: An Engineering Problem (CMU/SEI-93-SR-005, ADA267117). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, July 1993. Available WWW. URL: <http://www.sei.cmu.edu/publications/documents/93.reports/93.sr.005.html>.

Hammer,M. And Champy, J. Re-engineering the corporation – a manifesto for business revolution, Nicholas Brearly Publishing, London 1993.

Kullbach, B.; Winter, A. Querying as an enabling technology in software reengineering. Software Maintenance and Reengineering, 1999. Proceedings of the Third European Conference, pp.:42 - 50

Manzella, J., Mutafelija, B. Concept of re-engineering life-cycle, Systems Integration, 1992. ICSI '92., Proceedings of the Second International Conference. Pp. 566-571.

Premerlani, W., Blaha, M. An approach for reverse engineering of relational databases, Communications of the ACM, May 1994, pp. 42-9, .

Sneed, H. M.. Economics of software re-engineering. Journal of Software Maintenance: Research and Practice, Vol. 3, No. 3, pp:163-182, September 1991

Sneed, H.M. Planning the Reengineering of Legacy Systems. IEEE Software, 1995, Vol. 12, No. 1, pp. 24-34

7 Author CVs

Marcos Ruano-Mayoral

Marcos Ruano-Mayoral is a Research Assistant of the Computer Science Department at Universidad Carlos III de Madrid. He holds a BSc in Computer Systems from Universidad de Valladolid and a MSc in Computer Science from Universidad Carlos III de Madrid. He has been involved in several research projects as information management engineer and software consultant.

Inmaculada Puebla-Sánchez

Inmaculada Puebla-Sánchez is Faculty Member of the Universidad Francisco de Vitoria. She holds a BSc in Physics and a MSc in Telecommunications and Business Administration.

Ricardo Colomo-Palacios

Ricardo Colomo-Palacios has been a Faculty Member of the Computer Science Department at Universidad Carlos III de Madrid since 2002. His research interests include Software Process Improvement, Software Project Management and Information Systems. He received his PhD in Computer Science from the Universidad Politécnica of Madrid (2005). He also holds a MBA from the Instituto de Empresa (2002). He has been working as software engineer, project manager and software engineering consultant in several companies including Spanish IT leader INDRA.

Ángel García-Crespo

Angel García-Crespo is the Head of the SofLab Group at the Computer Science Department in the Universidad Carlos III de Madrid and the Head of the Institute for promotion of Innovation Pedro Juan de Lastanosa. He holds a PhD in Industrial Engineering from the Universidad Politécnica de Madrid (Award from the Instituto J.A. Artigas to the best thesis) and received an Executive MBA from the Instituto de Empresa. Professor García-Crespo has led and actively contributed to large European Projects of the FP V and VI, and also in many business cooperations. He is the author of more than a hundred publications in conferences, journals and books, both Spanish and international.

Managing Knowledge for Information Systems Evolution

Paolo Salvaneschi

University of Bergamo, Faculty of Engineering, Dalmine (BG), Italy
and Salvaneschi & Partners, Bergamo,Italy
pasalvan@alice.it

Abstract
The paper describes a method for managing the knowledge required during the evolution phase of an information system. The method applies a minimalist approach to the documentation and uses the concept of "Total Cost of Ownership". A document classification according to the document types "maps", "aspects" and "critical points" is proposed. They are organized into a common knowledge base subject to multiple views, each related to a class of users. The chosen technology for implementation is based on wiki tools that also integrate external specialized CASE tools. An industrial application related to the management of trucks flow inside a steel pipes plant of a large company is shown.

Keywords

Software evolution, Knowledge management, Information systems

1 Introduction

IT departments of non-ICT organizations typically manage both new projects and a platform of existing software applications running the company business processes. A new project runs for a limited time interval (for example one year), while the life of an existing application may be very long, for example ten years. During this long life, applications are subject to changes. People dealing with them (managers, users, analysts, programmers) require knowledge about these, in many cases large artefacts, composed by complex structures of code and data. It is quite common that, even if the initial development was done through a good development process, rooted on the best Software Engineering practices, the delivered documents become rapidly obsolete and essentially not useful. They where useful for understanding need, specify functions or design the software, but the cost of maintaining such a huge set of information during, for example, ten years is too high. People have no time to update the documents and there is no technology for updating them automatically. In addition, competent people may not be there when required, because they changed company years before. The problem is well known.

Manny Lehman [1] years ago, studied the "Software Evolution Laws" and Bennett [2] introduced the staged model of software lifecycle with the initial development being only the first stage. The following 'maintenance' phase is separated into an evolution stage followed by a servicing and phase out stages and each phase has specific characteristics. Nevertheless, if you see a Software Engineering manual used in university courses, the ninety percent of the material is related to the first year of the product life and the industrial practice still lacks of viable approaches for evolution support. In many cases the underlying idea, seems to be: "Do a good development during the first year and simply apply a good maintenance process for the remaining ten".

Our work is based on the idea that this assumption is wrong and that ten years evolution are intrinsically different, if compared with a development year. Designing things from scratch and evolving existing things for long time requires to manage different types of knowledge, has different people, time, cost constraints and requires different concepts, methods and tools.

In the paper, we concentrate on the specific problem of managing the knowledge required for supporting the evolution phase. We explore a specific set of questions: What type of knowledge is useful for supporting the evolution of long living software applications? How can we cope with tight cost and time constraints for keeping this knowledge alive? Where does this knowledge come from? Are there tools more suited for supporting the management of knowledge? Chapter 2 presents the exiting approaches and the contributions we tried to learn from. In chapter 3 and 4, we show the principles we based on our method and a classification of the types of knowledge. Chapter 5 discusses the use of Wiki based tools for supporting the knowledge management. All these concepts are then exemplified describing an application to an industrial case. Some conclusion and discussion about future works close the paper.

2 Existing approaches

The most common situation in industrial application of information systems is that, if several representations of software exist (e.g. specification, design, source code), only the source code is maintained, and the other representations become inconsistent and are no longer trusted. The problem of studying what types of representations are useful and may be maintained at a reasonable cost during the evolution phase has been poorly studied. Nevertheless, many approaches, mainly oriented towards the development phase, may provide useful suggestions.

Useful ideas come from the Agile Movement context [3,4]. They introduce interesting practices:

- Competent people is important;
- An "agile documentation" is not a "complete" documentation" but a "good enough";

- A document must focus on a specific and well-defined goal and must include practical and useful pieces of knowledge;
- The development of documentation should consider the concept of “Total Cost of Ownership”.

In this context, guidelines are also available for producing “lightweight” documents for the development phase [5], but the agile approaches are not specifically oriented towards the evolution phase.

Another useful idea is the “minimalist” method for the development of documentation [6,7]. The idea of "minimalist" documentation is based on the theory of minimalist instruction founded in the psychology of learning and problem solving. In this approach, the reader is offered with the minimal amount of information needed to get the task done. The minimalist approach has been also used in [8] for documenting software frameworks. This concept is important for the evolution because it recognises the principle of providing the minimum set of documents that can support the evolution task.

Moreover, the types of required documents depend on the classes of users. The research in the requirements analysis area developed the interesting method of "Viewpoint oriented analysis” [9]. Each viewpoint represents the specific abstractions exploited by a class of users for their specific goals. If we adapt this approach to the documentation for evolution support, we conclude that we need multiple sets of documents (possibly overlapping) for supporting different classes of users (for instance developers, system managers, help desk people).

A final important suggestion is coming from the “aspect oriented development” research area. The idea (see [10] for an overview) is that programming, but also design and specification, involves the understanding and modelling different “aspects” of a software application. For example, a design aspect may be a model of all the software modules involved in the “performance” non-functional property. This property may be not confined in a module, but may crosscut a number of them. This concept is important for the evolution, because the explicit modelling of important aspects may support specific important required changes.

All these methods provide useful suggestions but none of them is specifically oriented towards evolution support. In the following, we will present a method that tries to learn from them to develop a guideline for supporting the evolution of information systems.

3 Principles

The method we propose for the management of knowledge during the evolution phase of information systems is based on the following principles:

1. The knowledge produced and managed during the development phase is not the same knowledge needed during the evolution phase.
2. The knowledge is a property emerging from a system composed by experienced people and documents stored and managed through computer–based tools.
3. The concept of “Total Cost of Ownership” drives the search for the equilibrium point between the role of experienced people and documents.

Let us discuss the first principle. When we develop a new software application, we need to understand requirements, write specifications, design the architecture, detail design decisions and finally develop code and generate data structures. The aim is solving a synthesis problem. When we collect knowledge and we generate new knowledge, the result (code and data) is not there. This means that we need to generate a sequence of models from very abstract ones (for example a functional specification of an applications that will include a set of interactive form-based components and a data base) to very detailed ones. We prefer to use the term “model” emphasizing the meaning, than the term “document”. Models are written through diverse languages, from natural language to semi-formal or formal graphical representations and are embedded into documents or specialized tools. The resulting models may be, in this case, a detailed specification of each form, the design of the person/machine interface, the design of the internal structure of the person / machine interface and

the design of the data base. On the contrary, when we evolve an existing system, the synthesis problem (how to change the existing system) is just part of the job. A very significant part is solving an analysis problem. The product is there and you need to understand it. A number of documented design details may be no more necessary. A good programmer may deduce the design decisions reading the code. The detailed specification of forms, necessary in the negotiation with the users, is no more useful. You simply have to run the product and see the interface. Other models are certainly useful. For instance a model of the central data structure, explaining the meaning of data and the constraints on them or models of the user processes, if a business process is implemented through the execution of many forms in a constrained order. Other types of information, not so critical during the synthesis phase, are important. Examples are how to install the application or how to interface it with other applications. Other models describing specific aspects difficult to understand reading the code may be useful: for instance, a model describing all the software modules and the associated details related to the application of the security policy. Note that this model describes a specific aspect that may crosscut different software modules and may include high-level architectural descriptions as well as code ad data details. Not necessarily, the designer released this type of document that will be very useful if, in the evolution phase, a change in the security policy will be required. The rational is that just one part of the documents released by the development is useful during the evolution and additional documents may be specifically useful for this life phase.

The second principle says (obviously) that without experienced people the evolution is difficult. However, even if you have them in your organization, you can obtain bad performances if good models describing useful abstractions of the software are not there and the only available model is the final one: code and data. Not necessarily, models must be “complete” (rich of details in each part). It is sufficient that they be “good enough” for experienced people using them. Experienced people may use a model as a “landscape” for understanding where to read the details in the code. If the detail of models is excessive, people will not use and update them and they will rapidly become obsolete and will be abandoned. The rational is that the knowledge required for the evolution phase emerges from the interaction between people and software models and there is an “equilibrium area”, where knowledge hosted into people and into models co-operate efficiently.

The third principle guides the search for the “equilibrium area”. The idea is that the time spent by an experienced developer for understanding a software application is a cost. A model that helps the developer reduces this cost but requires an additional cost to be maintained up to date. The “Total Cost of Ownership” is the sum of these two costs. A strategy for optimising this cost may be:

- Identify the knowledge level of people involved in the evolution phase.
- Write a list of models whose absence may cause a significant improvement of the cost of understanding the application by the involved people and rank the improved cost in a scale “medium, high, very high”.
- Examine the list, estimate the maintenance cost of each model using the same scale. Use the two sets of costs for deciding if a model will be maintained or will be substituted by people knowledge. For instance, a model whose absence produces a “very high” cost of understanding and has a “medium” cost of maintenance is a good candidate to be used for the evolution phase.

The above-presented principles can be used for qualitatively driving the knowledge management process, but do not help the identification of what type of knowledge is useful for supporting the evolution of long living software applications. In the following, we will provide a first guideline for classifying this type of knowledge.

4 Knowledge classification

We propose a set of models used for supporting the evolution phase and based on the following classification:

- The models must support different views of the evolution phase. Each view is related to a specific class of users. A model may be specific for a view or may be used by multiple views. A view may use more models.

- Each view includes models belonging to three classes: “maps”, “aspects” and “critical points”.

The concept of “views” is because different classes of users may require support during the evolution phase. Typical classes are: software designers and programmers; system managers; user assistance people; analysts and marketing oriented people. Each class of users requires specific models. For instance, a designer may require a software architectural model, while an analyst may use a list of functions implemented by the product. More user classes may use a model. For instance, both designers and analysts may use a model describing the flow of information and control of a business process (how a set of functions are involved in the execution of the process).

Each “View” uses a set of models. For classifying the types of models, we exploited the Geographic Information System metaphor. A GIS helps people understand a complex system: an earth area composed of many interconnected parts and subsystems. Among them roads, urban areas, land use, rivers. The GIS provides tools for understanding this complex system. For instance, you can remove details and have a high level map helping you understand where you are and where is the place you want to reach. You can also select a specific aspect of your interest. For instance, you ask the GIS to visualize just the highways from the departure place to the arrival. You can also ask for specific information that is important for you like data of a Hotel. The idea of the metaphor is that a software application is a complex earth area and you have to develop GIS-like facilities for understanding it. Applying the metaphor, we identified models of type “maps”, “aspects” and “critical points”.

A model of the type “map” has the role of creating a landscape. A drawing of software architecture is a map. A programmer uses it for understanding a set of components, their roles and relations. The map helps locate what the programmer needs to modify. Additional details may be found reading the code. Updating the map must be mandatory (and audited) whenever the application is changed during the evolution phase. This is a cost item of the “Total Cost of Ownership”, but the cost of understanding the structure of the application without a “map”, when a change is needed, is much higher.

Aspects-based models describe a specific “aspect” (using the terminology of the aspect-oriented development) of the application. A model of this type crosscuts a set of software modules or functions. This set is specifically involved in the definition and implementation of a requirement (functional or non-functional) or is related to a specific goal. Examples are:

- The software modules and the related details contributing to the non-functional property “security”;
- The same for the property “safety”;
- A model describing the flow of control and information through a set of functions that implements a business process;
- A model describing a set of modules and the associated behaviour that is involved in a typical future request for change.

The third type of models is “critical points”. These are specific information related to important or particularly complex parts of the applications. We can see them as the annotations the programmer writes and maintains for avoiding known problems when changing code.

Note that this set of models is not a “complete view” of the software application. They are only the minimum set of important fragments of knowledge. The human part of the system (the competent people) links the fragments with the existing software implementation and generates a “complete enough” understanding for a specific purpose.

5 Using the WIKI tool

Another important problem to be considered is the technology to be used for implementing, maintaining and distributing the knowledge base. Our knowledge base has the following main characteristics:

- It is composed of fragments and each fragment (model) composing the knowledge base is unique, but it must be used through a rich structure of links. Moreover external CASE tools must be easily

linked;

- We want to enforce the idea that it is a living body of knowledge that is used and enhanced by a group of people.

For those reasons we used a Wiki tool as implementation environment. Wiki tools have been largely used for distributing knowledge (the Wikipedia). They have been also used for managing software documentation and integrating specialized CASE tools. See [8,11] as an example of documenting software products using a minimalist approach and a wiki tool.

6 An industrial example

In the following, we will show an example of an industrial application of the proposed method. The aim is the support to the evolution of software devoted to the management of trucks flow inside a steel pipes plant of a large company.
The application is in service from a number of years and manages the trucks flowing in and out the plant (access control, tracking, weighting, documents management, interaction with the management information system - more than 300 trucks/day). The software application size is about 100.000 lines of code and includes COBOL components, Java classes and ATP components (a special language similar to JSP). It runs on an HP Non-Stop fault-tolerant system with a proprietary DBMS in a UNIX like environment and interoperates with a variety of other systems (for instance SAP, AS400 and the weighting subsystems). The application had a long evolution history and progressively included a number of software technologies. The application is still subject to a significant evolution. Recently it has been adapted to be used in the context of a new plant in East Europe.

6.1 Views, maps, aspects and critical points

We defined four views: analyst, system manager, developer and user. The following table lists the implemented maps, aspects and critical points.

Maps		**A**	**S**	**D**	**U**
M1	Synthetic textual description of the application	X			
M2	List of functions	X			X
M3	Client Server architecture		X		
M4	Hardware components and interconnecting protocols		X		
M5	List of installed services		X		
M6	Software architecture (static view)			X	
M7	Data model			X	
Aspects					
A1	Petri Net describing a truck flowing in to be loaded	X			X
A2	Petri Net describing a truck flowing in and carrying a load	X			X
A3	Petri Net describing a generic truck flow	X			X
A4	Petri Net describing a railway mean flowing in to be loaded	X			X
A5	Petri Net describing a railway mean flowing in and carrying a load	X			X
A6-10	A1 to A5 models annotated with the involved hardware systems		X		
A11-15	A1 to A5 models annotated with the involved software components			X	
Critical points					
P1	List of log files		X	X	
P2	Notes on management of the RTA service		X	X	
P3	Specific algorithms and business rules			X	
P4	Frequently Asked Questions				X

Table 1: An example of knowledge for evolution

The table also shows the relation between models and views (the letters A, S, D, U stand for Analyst, System manager, Developer, User). The set of Petri Nets (from A1 to A5) model the business processes implemented by the application. Each model describes the flow of control through a set of functions (a functional "aspect"). In the models A6-10, each involved function is annotated with the information of the hosting system. Many hardware systems cooperate for delivering the business process and the model is useful both for supporting the user assistance in case of malfunctions and for supporting the change. Models A11-15, implement the same approach but the annotation is related to the software components implementing the considered functions.

6.2 Wiki implementation

All the views and the models above listed have been implemented using the Deki Wiki tool. Fig 1 shows a couple of screens of the implementation.

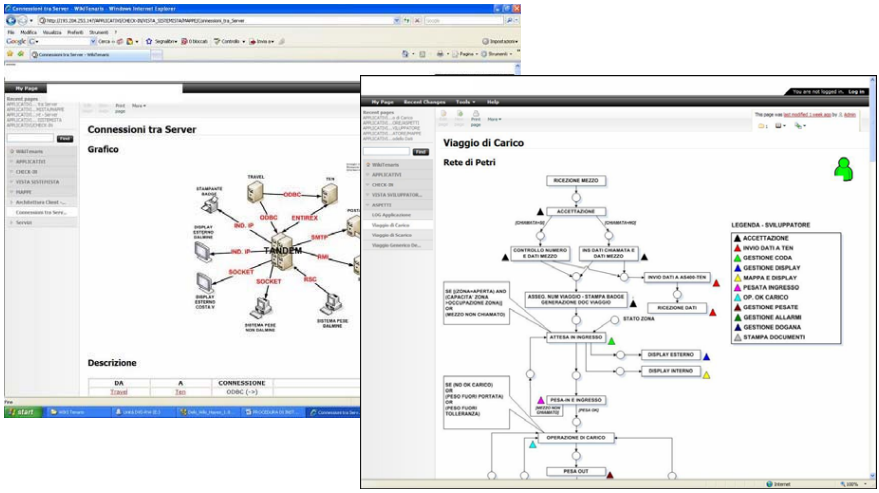


Figure 1: The wiki implementation of knowledge for evolution

The left side page is a "map" type model for the view "System Manager". The model includes a drawing of system components and protocols. Additional information for each hardware component follows below. According to GIS metaphor, this model is a "landscape" for the system manager. It is used for understanding the set of involved hardware components, their main characteristics and relations.

The right side page is an "aspect" for the developer. It is a Petri Net modelling the flow of control between different functions of the application for the business process "truck flowing in to be loaded". The icons on the side of each function define, through different colours, the software components where the function is implemented. Comparing to a GIS map, this is an interesting aspect crosscutting the available geographical information (like for example a highway route) The model provides useful information for driving a change request for the business process or for locating causes when processing a problem ticket concerning the business process.

Not all the models are included into the wiki tool. For instance, the data model of the central database (a "map" type model) is managed through a link to an external CASE tool.

The system is now delivered to the users. The development team is completing some models. The initial experience of use shows that the most interested groups of users are the system manager and

the developers. They use the wiki tool as troubleshooting support. The stored knowledge has been also useful during the development of the new version for the new plant in East Europe.

7 Conclusions

The aim of the method and the related industrial experience is to support an important issue in the area of information systems that, until now, has been poorly addressed. The industrial example is an on-going experience and more work has to be done to assess the value of the proposal. Nevertheless, the method provides a first approach for classifying and managing the required knowledge. An interesting line of improvement is the classification of types of software applications and the definition of specific patterns for maps, aspects and critical points.
An important element of the method, not specifically addressed by the paper, is the process associated to the management of the knowledge base. The essential points to be considered are:

- Provide guidelines for the developers so that a significant part of models required for the evolution will be available at the delivery time;
- Define an audit phase before the delivery to check the models and add additional models;
- Deliver both the application and the Wiki knowledge base;
- Define the ownership and the roles associated to the management of the knowledge base.

The method has to be further tested, extending the complexity of the considered software applications. More experiences are now running in two different contexts: the evolution of a product line for CAD-CAM-CIM applications and the evolution of a large information system for a retail company.

8 References

[1] Lehman M. M. Program evolution. Academic Press, London. 1985.
[2] Bennett, K.H., Rajlich, V.T., 2000. Software Maintenance and Evolution: a Roadmap. In The Future of Software Engineering, A. Finkelstein Ed., ACM Press, New York, NY.
[3] K. Beck. Extreme Programming explained: embrace change. Addison-Wesley, 1999.
[4] S. W. Ambler. agile modelling. John Wiley & Sons, 2002.
[5] A. Rüping. Agile Documentation - A Pattern Guide to Producing Lightweight Documents for Software Projects. John Wiley & Sons, 2003.
[6] Carroll, J. M. - The Nurnberg Funnel: Designing Minimalist Instruction for Practical Computer Skill. MIT Press - 1990.
[7] Van Der Meij, H. and Carroll, J. M. - Principles and heuristics for designing minimalist instruction. In Minimalism Beyond The Nurnberg Tunnel - 1998.
[8] Teixeira de Aguiar A. M. – Framework Documentation A Minimalist Approach. Universidade do Porto, PhD Thesis, 2003.
[9] Darke, P. & Shanks, G. (1996). Stakeholder Viewpoints in Requirements Definition: A Framework for Understanding Viewpoint Development Approaches. Requirements Engineering,1(2): 88-105.
[10] AOSD-Europe, Survey of Aspect-Oriented Analysis and Design Approaches. www.aosd europe.net, 2005.
[11] Teixeira de Aguiar A. M. – WikiWiki Weaving Heterogeneous Software Artifacts. WikiSym '05, San Diego, CA, U.S.A. 2005 ACM

9 Author CVs

Paolo Salvaneschi
Associate professor of "Software Engineering" at University of Bergamo, Faculty of Engineering and director 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 and A.I. application to monitoring and data interpretation as well as ICT applications in various areas (industry, civil engineering, environmental control).

Integrated Engineering Skills: Improving your System Competence Level

Andreas Riel[1], Serge Tichkiewitch[1], Richard Messnarz[2]
[1]Grenoble Institute of Technology (France) and EMIRAcle AISBL
[2]ISCN GmbH (Austria) and ECQA
Andreas.Riel@inpg.fr, Serge.Tichkiewitch@inpg.fr, rmess@iscn.com

Abstract

Integrated Product Development is increasingly a challenge of understanding and predicting the complete product lifecycle, and of treating the product as a whole in all phases of development. This paper points out that this trend has significant implications on the competence profiles of engineers, which are nowadays insufficiently taken into account by education and training programs, as well as skill certification schemes. Building on the highly innovative and successful European-wide professional training and certification scheme of the European Certificates and Qualification Association (ECQA), it suggests a basic skill profile that is characteristic for modern job roles in Integrated Engineering, and justifies the integration into the ECQA platform. It points out that this initiative marks a major step towards the improvement of engineers' system competence levels.

Keywords

Integrated Engineering, Integrated Design Engineer, System Competence, Product Development Improvement, Lifelong Learning, Certification, Professional Training

1 Introduction

Integrated Engineering is characterised by a highly multidisciplinary approach to product development. 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. Likewise, engineering teams are getting increasingly interdisciplinary, and thus demand for a mutual understanding and collaboration between domain expert team members [23][26].

Although university curricula are starting to get adapted to this development on an international scale, it is evident that there is an urgent need for interdisciplinary education and certification programs on a postgraduate level. While universities are supposed to educate in-depth knowledge in specific engineering areas, lifelong learning programs and curricula are needed that teach the transversal links between the different engineering disciplines according to criteria that are defined by industry. Industrialists demand for the certification of these skills, as well as for their international recognition and exchangeability.

Today, such internationally recognized training and certification programs for job roles in modern manufacturing do not exist. This paper describes the approach that EMIRAcle (the European Manufacturing and Innovation Research Association, a cluster leading excellence – http://www.emiracle.eu) takes together with the ECQA (the European Certificates and Qualification Association – www.eu-certificates.org) 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 skill sets that

characterises Integrated Engineering, as well as to provide skill-specific training modules and the corresponding training material. Once these are found, test questions have to be formulated, which shall provide the basis for assessment and certification of candidates.

We present the requirements to job roles in Integrated Engineering that are demanded by industry, and show how they are used to develop education and test programs, as well as certification criteria. This activity is part of the EU Certification Campus (EU Cert) initiative launched by the ECQA at the beginning of 2008, which aims at implementing a number of training and certification programs into their already well-established IT-platform, and offering those in a number of education institutions all over Europe. While ECQA members are already very active and successful in the software domain, EMIRAcle will extend their concept to the manufacturing domain with a string focus on innovation aspects. The work presented here is thus the basis for a life-long learning Integrated Engineering training and certification concept and infrastructure which are both not only worldwide unique but also heavily demanded by the industry [20].

Chapter 2 summarizes the background of the work of the ECQA. In Chapter 3 we point out the requirements to Integrated Engineering skills and we make evident that those are not sufficiently taken into account in current education schemes. Chapter 4 links aspects of improvement to system competence. Chapter 5 introduces the training, testing and certification concept that has been established by the ECQA and suggests it as the platform to implement job roles in Integrated Engineering.

2 Background

This section points out the need for job role based qualification and certification in industry.

2.1 Success Factors of Innovation

The success of an innovation or improvement is not just dependent on the correct technical approach. Instead, numerous learning strategy related aspects influences the success. This fact has been proved by the following European studies, among others:

- Study at 200 firms in 1998 [15];
- study at 128 multinational firms in 2002 [18];
- study in 59 networked European organisations in 2003 [2][8][9].

Beside top management support (26%) the studies outlined a positive learning culture (15%, learning from mistakes, team learning, knowledge sharing, etc.) and a supporting organizational infrastructure (17%) which helps with the implementation of the learning organisation [18]. A learning organisation [10][17] creates a positive learning culture and enables team learning and synergy exploitation in an organisation. By team learning knowledge is spread much more quickly and a high level of a skilled human force is maintained.

Human skills are regarded as a complementary set needed in addition to qualified processes to be successful on the market.

2.2 Processes, Job Roles, and Skills

Figure 1 illustrates that processes require roles, which need specific skills to efficiently perform the job. In ISO 15504 a capability level 3 would, for instance, require the definition of competence criteria per role. The combination of this approach with the learning organisation related approach outlined in section 2.1 leads to a framework where it becomes extremely important to think in terms of job role based qualification and skills. This concept is described in greater detail in e.g. [14].

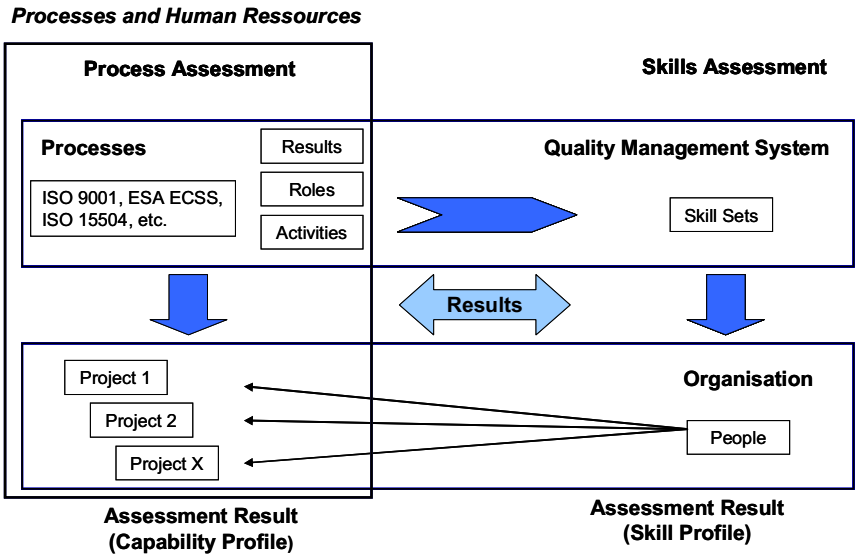


Figure 1. Integration of Process and Human Skills in an Integrated Model

3 Key Skills of Engineers in Integrated Engineering

In this section we suggest four of the key skill units which we believe should complement expert skills for job roles Integrated Engineering, with a strong focus on Integrated Design as design is at the root of every product development. The skill sets that make up this unit, as well as additional units will be developed in the frame of this research project.

3.1 Requirements Engineering

The key to making a product successful on the market is to design it according to all sorts of key requirements that come from a number of different sources. These are all the actors directly involved in the product life cycle, as well as the product's "environment", like government, laws, economy, etc. Outstanding actors and factors are

1. the target customers,
2. the manufacturing process,
3. the product's life cycle,
4. its manufacturability and maintainability,
5. the development time and costs,
6. etc.

Identifying requirements is in general a complex activity. Very often the requirements specifications that are given to designers are imprecise and/or incomplete. Knowledge about systematic requirements collection and management helps designers collect missing or incomplete requirements information.

Requirements management is a complex procedure that is difficult to carry out systematically without the use of appropriate tools. There exists already a large number of requirements management tools

(about 40 are listed in [12]), which are typically specialized for use in certain domains. Even if in some (especially bigger) organizations development tools are chosen on a higher management level, it is often the engineers who are asked to propose a choice of tools.

User-centric methods like Scenario-Based Design [6] and Use Case Design [3] are becoming more and more important, as they force requirements engineers to think from a product-use point of view rather than in terms of solutions. Scenarios are important tools for exercising architecture to gain information about a system's fitness with respect to a set of desired quality attributes.

A use case is a description of a system's behaviour as it responds to a request that originates from outside of that system. The use case technique is used in software and systems engineering to capture the functional requirements of a system. Use cases describe the interaction between a primary actor—the initiator of the interaction—and the system itself, represented as a sequence of simple steps. Actors are something or someone which exist outside the system under study, and who (or which) take part in a sequence of activities in a dialogue with the system, to achieve some goal: they may be end users, other systems, or hardware devices. Each use case is a complete series of events, described from the point of view of the actor.

Use case design thus enables engineers and anyone else concerned with the product to adopt an application and user-oriented viewpoint which largely facilitates the derivation of the detailed functional requirements to the product.

3.2 Integrated Product Design

Current design methodology developed a lot of tools called “Design for …”, in order to take into account one specific domain (assembly, maintenance, manufacturing, etc.). Such tools are made to optimize one specific view, disregarding the fact that the global optimization of a system is in general not to be achieved by the local optimization of a series of components. Moreover, what normally has to be a constraint for the system is transformed into an objective function in these systems: Does an assembly have to be minimized, or is it sufficient to respect its operability if in another solution it can be less costly or complicated?

Integrated product design considers that the different constraints previously cited are the aim of different actors who have to control them but who “belong to the same world” [4]. The common goal is to reduce the cost, to reduce the time to market, to take into account sustainability and to increase quality. Such actors have to work in a concurrent engineering context, having access to a common product model where they can have their own contextual views. They have to respect the just need [5] which consists of giving a constraint on the system as soon as possible if such a constraint can be proved.

An application of integrated design of wood furniture can be found in [19]. It is shown how the actors of the design process have to exchange information before starting a new design in order to understand what the consequences are of the different decisions they have to make for the other actors and which information has to be propagated. Choosing an assembly system for joining two boards is directly guided by a quality requirement but also has consequences on the mechanical models used to determine the deflections of the boards and on the manufacturing features to be realized (and therefore also on the cost). The assembly set can be considered as an intermediate object for the communication between people in charge of assembly, mechanical behaviour and manufacturing. As such it acts as a vehicular object (as opposed to a vernacular one). At the same time, however, this assembly set cannot be sized without knowing the thickness of the board that depends on the model used. We saw that an interactive process between the assembly actor and the people in charge of mechanics must arise during the design activity. This interactive process is a way to solve imaginary complexity.

Other particularly representative confirmations and urgent demands of the above issues have been published notably in the automotive industry [13][20], where product development is outstandingly multidisciplinary and interdependent.

According to the above, product design does not seek to optimize one single objective, but rather aims at finding the best compromise solution under multiple, often coupled restrictions like the following:

- Producability,
- Assembly/Disassembly,
- Modularity,
- Testability,
- Product Variant Creation,
- Environmental Sustainability,
- Product-Service Optimization,
- Maintainability,
- Cost Minimization,
- etc.

Certainly an Integrated Design Engineer cannot master all the associated complex disciplines by himself in general. He should, however, be able to understand domain experts, and be able to translate their requirements into his design task.

3.3 Product Lifecycle Engineering and Management

Integrated Engineering is synonym for well understanding the product and the way it is created, used, disposed, and recycled. Product Lifecycle Management (PLM) is the process of managing the entire lifecycle of a product from its conception, through design and manufacture, to service and disposal [24]. It is one of the four cornerstones of a corporation's information technology structure. All companies need to manage communications and information with their customers (CRM-Customer Relationship Management) and their suppliers (SCM-Supply Chain Management) and the resources within the enterprise (ERP-Enterprise Resource Planning). In addition, manufacturing engineering companies must also develop, describe, manage and communicate information about their products (PDM).

Although a product lifecycle is specific to a product, there are some basic facts, aspects, and phases that are common to almost any type of product. An Integrated Design Engineer needs this basic knowledge in order to be able to analyse and understand specific product lifecycles.

The core of PLM is in the creation and central management of all product data and the technology used to access this information and knowledge. PLM as a discipline emerged from tools such as CAD, CAM and PDM, but can be viewed as the integration of these tools with methods, people and the processes through all stages of a product's life. It is not just about software technology but is also a business strategy.

For simplicity the stages described are listed below in a traditional sequential engineering workflow. The exact order of event and tasks will vary according to the product and industry in question but the main processes are:

- Conception,
- Specification,
- Concept Design,
- Design,
- Detailed Design,
- Validation and Analysis (Simulation),
- Tool Design,
- Realization,
- Plan Manufacturing,
- Manufacturing,
- Build/Assembly,
- Test (Quality Check),
- Service,
- Selling and Delivery,
- Usage,
- Maintainance and Support,
- Disposal and Recycling.

The reality is however much more complex, people and departments cannot perform their tasks in isolation and one activity cannot simply finish and the next activity start. Design is an iterative process, often designs need to be modified due to manufacturing constraints or conflicting requirements.

Although Collaborative Engineering is based on the support of the organization, it is very much facilitated by the awareness of each engineer about his role in the process, as well as the roles of others.

3.4 Networked Collaboration

Due to the involvement of many different experts, Integrated Product Design can only be done in teams, which are inherently heterogeneous and also increasingly international. Although design tools support this collaborative work increasingly better, Integrated Design Engineers need to have skills that go beyond tool operation in order to be successful collaborative engineering tasks. We list below the ones that we will focus on in the development of the Integrated Design Engineer's profile:

1. Teamworking skills,
2. Intercultural skills,
3. Knowledge Management,
4. Knowledge Capitalisation,
5. Knowledge Sharing.

Teamworking and intercultural skills are indispensible in modern international engineering teams. Knowledge management is certainly a subject of the whole organisation, which is under the responsibility of the management levels. Understanding the purposes and challenges of knowledge management and knowledge capitalisation, as well as the concept of typical knowledge management and knowledge modelling tools, is an important prerequisite for the participation of Integrated Design Engineers in the related efforts of an organisation [7].

4 Improvement of and by System Competence

Integrated Engineering by its very definition covers multiple expert domains and thus usually separate and specific threads of communication, specific wordings, different understandings of terms, etc. Classic product development organisations typically resemble expert domains in their departmental and/or project structures, thus further intensifying and augmenting the difficulties of realizing integrated engineering. With increasing system complexity, obtaining the competence of the whole final product as a system and as a result of a networked system of development tasks has become practically impossible in such environments.

The development process of automotive powertrains is a stereotype example for this problem. The automotive industry is one of the most highly innovation-driven industries. This chapter presents selected results of a detailed analysis of this process [20], and their implications on the need for integrated engineering skills to attain and improve system competence.

4.1 Example: The Automotive Powertrain Development Process

Figure 2 shows the most essential phases of the powertrain development process [20]. The engine and transmission development processes run in parallel in very similar phases and they are closely linked by consecutive "vertical" tasks if the powertrain is developed in a holistic way. The horizontal line arcs indicate the various horizontal activities that need to be carried out ideally throughout the whole process, as they are all closely linked to the performance and quality of the final product. Most of them, however, require the whole powertrain and/or the vehicle to be available before they have actually been built. This is especially true for the engine and powertrain electronic control units (ECU – Engine Control Unit, TCU – Transmission Control Unit). In the traditional approach, prototypes of the missing parts are manufactured, or they are used from a suitable predecessor model.

In the modern, still heavily researched approach, simulation models with different levels of detail are used to mimic real components that are not yet available, from concept simulation via tests and calibrations on various kinds of testbeds to the phase with the vehicle prototype on the chassis dynamometer. This enables "front-loading" development activities to the early phases of the process, which are mostly linked to design. In this scenario, it may well happen that the transmission exists before the engine has been built and vice versa.

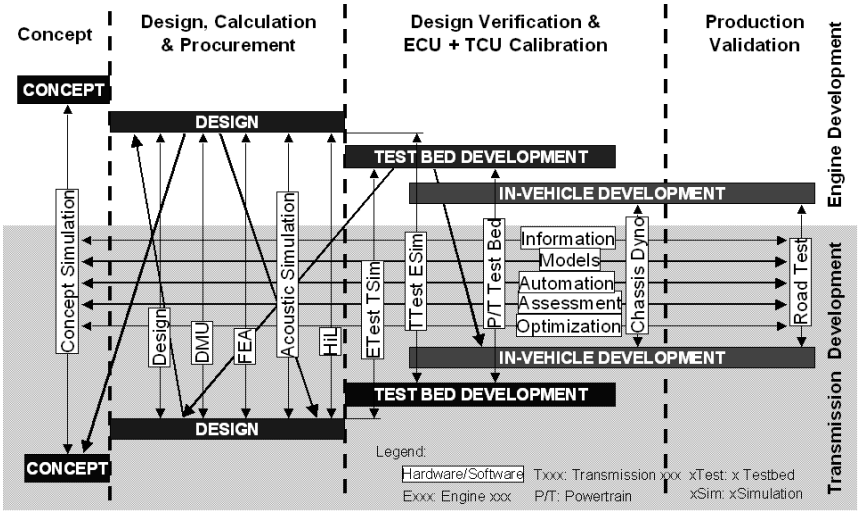


Figure 2. . Automotive Powertrain Development Process

Both these approaches, and any approach in between, represent cases in need of intensive integrated engineering and system competence on an individual engineer's level as well as on a distributed team level. They involve engineers with several different education and expertise profiles, who all have to work towards the same final targets, which are all linked to the global performance of the whole vehicle, mainly in terms of drivability (specific "feeling"), fuel consumption and emissions. The inputs of one activity depend on the results of several other activities, which are all linked to different domain experts. [5][7][26] treat this subjects exhaustively, with special regard to its implications on integrated design. [20] develops the so-called Behavioural Mock-Up (BMU) concept that extends the well-established Digital Mock-Up (DMU) concept to support the entire development process.

The permanent interactions and synchronizations between the two processes are sketched with the inclined arrows in Figure 1. Networking the engine and transmission development processes can be achieved by the seamless use of simulation tools and consistent simulation models. Closely connected to this is the process of collecting all the data that are required for the models used [20]. Primarily due to the stringent demands imposed by *quality assurance*, member of the different, typically distributed engineering teams, need to have comparable levels of engineering skills on a *system level*. Because it is on a system level where the teams' tasks are linked and have their dependencies: Engine and transmission, control electronics and powertrain, comfort electronics and cabin, etc. to name only a few.

4.2 Model-Based Integrated Development

In the ideal model-based integrated development process, sketched in Figure 3, the early CAE-models

act as the single source of data for all the later models. This assures the consistency of all the models.

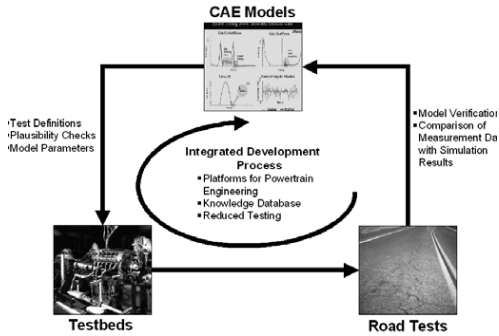


Figure 3. Model-based Integrated Development

Real-time models are derived from CAE-models by target-oriented simplification or re-structuring, which typically includes the replacement of analytical calculations by pre-calculated maps and the exclusive use of fixed-step solvers. CAD/CAE data and models are used for test planning and definition, and a seamless feedback loop from the testing environment has to be established for model verification and improvement. A practical example can be found in [21]. This engineering "control loop" relies on a working flow of vehicular knowledge between the involved groups and departments. Realizing such a loop relies on system competence of the engineers involved: Each part of this loop has to understand what the other parts need in terms of the characteristics of the system models, the parameters and the data.

4.3 IT-Infrastructure in Integrated Engineering Organizations

A fundamental requirement to integrated engineering support systems is to neatly integrate into existing IT infrastructures. Both manufacturers and suppliers have invested a lot in their tool- and IT-infrastructures. CAD, ERP and PDM systems are more or less the three IT "pillars" within a product development enterprise [20]. Figure 4 shows the close relationships between the integrated engineering environment (here represented by the BMU) and all the other important complementary information sources within the enterprise.

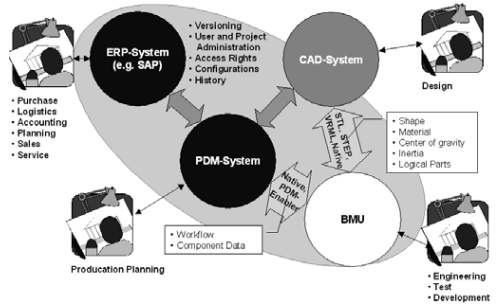


Figure 4. Networked Integrated Engineering

Integrated Engineers have to understand the role of each system in order to be able to use the whole IT infrastructure in way that it can leverage the work of all the concerned engineering teams. Once more he has to be aware of the fact he is one part in a highly networked, dependent and complex

system, in which his work depends on that of others and vice versa [25].

5 Qualification and Certification of Integrated Engineering Skills

We are fundamentally convinced that the system and the platform proposed and implemented by the ECQA [14] is very well suited to specify, implement and roll out the qualification and certification of modern job roles in Integrated Engineering environments.

5.1 Skills Acquisition with the ECQA Platform

The ECQA has set up a partnership of experienced partners in 18 European countries to create a pool of knowledge for specific professions. This pool can be extended to further professions. All the professions that have been configured in this system up to now, are based in the ICT area, and are thus closely related to Software Development. As integrated product development processes are increasingly related and/or linked to software development, we believe that new job roles from the Integrated Engineering domain will profit from this basis [22].

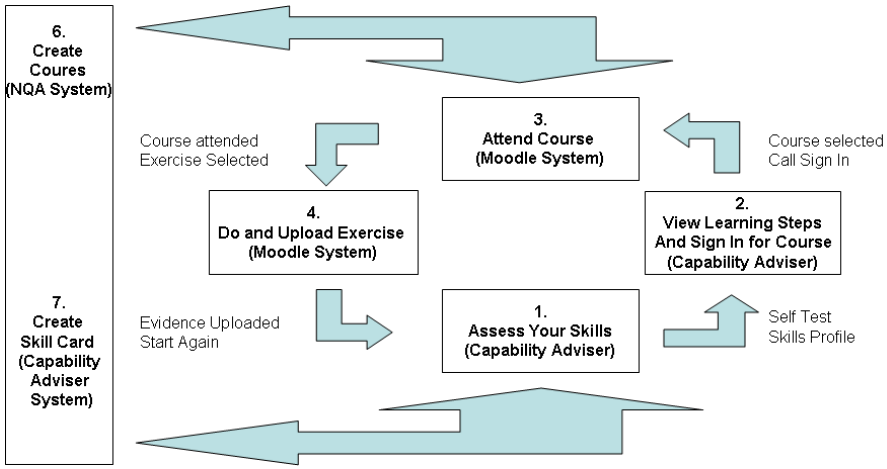


Figure 5. The Integrated European Skills Acquisition System

Figure 5 gives an overview of the uncomplicated but efficient skill acquisition process supported by the ECQA platform: If there is a need a person can attend a course for a specific job role online through an advanced learning infrastructure. The student starts with a self assessment against the skills [16]. Then she can sign into an online course. Here he is guided by a tutor and does a homework which is corrected by the tutor. Finally the homework and the real work done in her project are sufficient to demonstrate the skills.

The learning platform is based on the web based public domain learning management system Moodle (www.moodle.com). The assessment process is supported by the so-called Capability Adviser, which is a web based assessment portal system with a defined database interface to connect the systems. Network Quality Assurance NQA is a web based team working tool which was developed in the EU IST 2000 28162 project [16].

5.2 Provision of Skill Sets

The ECQA platform of knowledge is enhanced on an annual basis. Existing skills sets are being re-worked and new skills sets are added. Joint knowledge is being configured in form of a job role with standard content structures [8][17] like skills set, syllabus, learning materials and online configuration, as well as sets of test questions.

So-called Job Role Committees decide upon the content for a specific skills set. These committees are composed of academics and industrialists. The job role committee for the Innovation Manager, for instance, created a skills set of an innovation manager together with a set of online courses etc. People can register from their work places.

5.3 Qualification and Certification

Nowadays and according to the Bologna Process, it is very important that training courses are internationally recognized, and that successful course attendees receive certificates that are valid for all European countries. The EU supported the establishment of the European Qualification Network (EQN), from which the ECQA has evolved, with exactly this target in mind.

This has resulted in a pool of professions in which a high level of European comparability has been achieved by a Europe wide agreed syllabus and skills set, a European test questions pool and European exam (computer automated by portals) systems, and a common set of certificate levels and a common process to issue certificates.

The partners collaborated on the development of the quality criteria consisting of: Quality criteria to accept new job roles in the ECQA, quality criteria to accredit training organisations and certify trainers promoted by the ECQA, and quality criteria and test processes to certify attendees who have run through the training of a specific job role.

The existing skills assessment portals (already used by more than 5000 students in different learning initiatives) are extended to cover the new requirements of the ISO 17024 (General Requirements for Bodies operating Certification of Persons) standard. Among the international certification organizations that provide ECQA-compliant certification is the ISQI (International SW Quality Institute, www.isqi.org).

6 Conclusion

This paper points out that there is a strong industry need for international training, qualification and certification of modern job roles in Integrated Engineering. We found that the lifelong learning concept of the ECQA, which is already very well established in the ICT domain and set a European-wide standard there, is highly suitable for this purpose. Moreover, the ECQA provides a strong IT platform with all the applications required for learning, testing, and certification already in place. We have started related activities for two specific job roles which are already very demanded in various industrial sectors: Integrated Design Engineer and Innovation Manager in Engineering and Production.

Acknowledgements

This project is the first in the long-term lifelong learning strategy of the EMIRAcle association, which is well-aligned with the strategic objectives of the European Technology Platform in Manufacturing ManuFuture (www.manufuture.org). 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. We are currently supported by the EU in the Leonardo da Vinci project LLP-1-2007-AT-KA3-KA3MP (EU Cert - EU Certification Campus) of the Lifelong Learning Program.

Literature

1. Ameri F., Dutta D., Product Lifecycle Management: Closing the Knowledge Loops, Computer-Aided Design & Applications, Vol. 2, No. 5, 2005, pp 577-590
2. Biro M., Messnarz R., Davison A., The Impact of National Cultures on the Effectiveness of Improvement methods – The Third Dimension, in Software Quality Professional, Volume Four, Issue Four, 2002, American Society for Quality
3. Bittner K., Spence I., Use Case Modeling. Addison Wesley Professional, 2-3. ISBN 0-201-70913-9, 2002
4. Boltanski L., Thevenot L., De la justification, les économies de grandeur, Gallimard, 1991
5. Brissaud D., Tichkiewitch S., "Innovation and manufacturability analysis in an integrated design context", in Computers in Industry, 43, 2000, pp 111-121
6. Carroll J.M., Five Reasons for Scenario-Based Design, in: Proceedings of the 32nd Hawaii International Conference on System Science, Hawaii, IEEE No. 0-7695-0001-3/99, 1999
7. Draghici G., Brissaud D., Modélisation de la connaissance pour la conception et la fabrication integrées, Editura Mirton, Timisoara, 2000
8. Feuer E., Messnarz R., Best Practices in E-Commerce: Strategies, Skills, and Processes, in: Proceedings of the E2002 Conference, E-Business and E-Work, Novel solutions for a global networked economy, eds. Brian Stanford Smith, Enrica Chiozza, IOS Press, Amsterdam, Berlin, Oxford, Tokyo, Washington, 2002
9. Feuer E., Messnarz R., Wittenbrink H., Experiences With Managing Social Patterns in Defined Distributed Working Processes, in: Proceedings of the EuroSPI 2003 Conference, 10-12 December 2003, FTI Verlag, ISBN 3-901351-84-1, 2003
10. Gemünden H.G., Ritter T., Inter-organisational Relationships and Networks, Journal of Business Research, 2001
11. Hua C., Chao L., Yan C., Rujia Z., Xu Q., Co-operative design based on multi-agent systems. In: Cheng K., Webb D., Marsh R. (eds.): Advances in e-Engineering and Digital Enterprise Technology: Proceedings of the 4th International Conference on E-Engineering and Digital Enterprise, Wiley, 2007, Wiley, 2007
12. Ludwig, J.I., Requirements Management Tools, available at: http://www.jiludwig.com/ Requirements_Management_Tools.html, Accessed: 2008-04-14
13. Menne R., Ford: teach integrated engineering, in: Automotive Engineer, December 2007, Professional Engineering Publishing Limited, London, UK, p. 5, 2007
14. Messnarz R. et al., The EQN Guide. Graz, Austria, 2008
15. Messnarz R., Stöckler C., Velasco G., O'Suilleabhain G., A Learning Organisation Approach for Process Improvement in the Service Sector, in: Proceedings of the EuroSPI 1999 Conference, 25-27 October 1999, Pori, Finland, 1999
16. Messnarz R., Stubenrauch R., Melcher M., Bernhard R., Network Based Quality Assurance, in: Proceedings of the 6th European Conference on Quality Assurance, 10-12 April 1999, Vienna, Austria, 1999
17. Messnarz R., Nadasi G., O'Leary E., Foley B., Experience with Teamwork in Distributed Work Environments, in: Proceedings of the E2001 Conference, E-Work and E-commerce, Novel solutions for a global networked economy, eds. Brian Stanford Smith, Enrica Chiozza, IOS Press, Amsterdam, Berlin, Oxford, Tokyo, Washington, 2001
18. O'Keeffe, T., Harrington D., 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, 2001
19. Pimapunsri K., Tichkiewitch S., Butdee S., "Collaborative negotiation between designers and manufacturers in the wood furniture industry using particleboard and fibreboard", Design Synthesis, CIRP Design Conference, Enschede, CD-ROM, 2008
20. Riel A., From DMU to BMU: Towards Simulation-Guided Automotive Powertrain Development. Thèse de Docteur Ingénieur, Université de Technologie de Vienne, Vienna, Austria, 2005
21. Riel A.; Brenner E., Shape to Function: From DMU to BMU, in: Experiences from the Future. New Methods and Applications in Simulation for Production and Logistics. Fraunhofer IRB Verlag, Stuttgart , ISBN 3-8167-6640-4, 2004, pp 275-288, 2004

22. Riel A., “EU Certificates and Knowledge Communities in Europe: An unbeatable Symbiosis”, Keynote at “EQN Founding and Dissemination Conference”, Krems, Austria, CD-ROM, 2006

23. Riel A., Tichkiewitch S., Molcho G., Shpitalni M., Uys W., Uys E., du Preez N., Improving Product Development Organisations using Knowledge Mining: Requirements, Methods and Tools. Knowledge Management in Product Development, Proceedings, Enschede, CD-ROM, 2008

24. Saaksvuori A., Product Lifecycle Management. Springer, ISBN 3540257314, 2005

25. Schmidt C., Temple B.K., McCready A., Newman J., Kinzler S.C., Virtuality-based understanding scheme (VirUS) – a holistic typecast supporting research and behaviour-oriented management of virtual teams. In: Cheng K., Webb D., Marsh R. (eds.): Advances in e-Engineering and Digital Enterprise Technology: Proceedings of the 4th International Conference on E-Engineering and Digital Enterprise, Wiley, 2007

26. Tichkiewitch S., Brissaud D., Methods and Tools for Co-operative and Integrated Design, Kluwer, Academic Publishers, ISBN 1-4020-1889-4, 2004, pp. 488.

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; Andreas.Riel@inpg.fr

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

Experiences in Training for Software Process Improvement

Christiane Gresse von Wangenheim[1, 2], Jean Carlo R. Hauck[1, 2], Richard H. de Souza[2], Maiara Heil Cancian[2]
[1]Universidade do Vale do Itajaí (UNIVALI) – Computer Science
São José/SC – Brazil
gresse@gmail.com
[2]Federal University of Santa Catarina (UFSC)
Florianópolis/SC – Brazil
jeanhauck@egc.ufsc.br, richardhenrique@gmail.com, maiara@telemedicina.ufsc.br

Abstract

An important factor for successful software process improvement are people, which need to be motivated and adequately trained to perform the process(es) being improved. And, although, there exist various software process improvement approaches, they, generally, do not provide detailed information on when to train what to which level of detail using which methodology. This paper reports our experiences and lessons learned in designing a tailored training program throughout a software process improvement initiative at the CYCLOPS Group at the Federal University of Santa Catarina/Brazil.

Keywords

Training, Software Process Improvement

1 Introduction

Successful software process improvement requires an effective alignment of people, processes and technology to establish process improvements that meet business goals. An important factor are people, which need to be motivated, committed and have the necessary competency to perform adequately the process(es) being improved. Even, although process performers may be competent to execute the process(es) in place, they may - especially in more immature organizations - lack knowledge on specific process areas, such as, e.g., requirements management. Inadequate knowledge can misdirect SPI efforts and, in practice, various organizations abandon their software process improvement (SPI) initiatives, because inappropriate training led to negative outcomes, or, in many cases, no implementation of proposed solutions [6, 11]. Thus, for SPI initiatives to succeed, organizations must properly align training to meet this need [6]. In practice, especially when considering SPI initiatives in small organizations, SPI training is often provided in an inadequate way, which does not satisfy the specific knowledge needs. For example, in many collaborative SPI initiatives, involving several small organizations at the same time, training is typically provided in blocks with a general training unit on SPI and reference models and then one unit about each process area (e.g., project planning or requirements management). In our experience, such training programs do not achieve their goal. Often the content of such trainings is too theoretical, too broad or too specific and does not provide concrete examples within the context of a specific organization. Especially, when considering that throughout a software process improvement program, there exist diverse training needs for different roles involved, e.g., basic understanding for managers in the beginning, process performers on the organizational standard process, etc., such a block design does not seem to be ideal. And, as it may be the only form

of training provided, typically, more people than necessary may be involved, consuming considerable effort spent in training [8]. Another factor is that such a training may be provided not related to the actual schedule of the SPI program and, thus, the delay in applying the acquired knowledge may contribute negatively on the learning effect and motivation.

Yet, especially small organizations often have to surge into inadequate off-the-shelf training programs, which often fail to teach people effectively on how to do their individual activities. And, although, there exist various software process improvement approaches, they, generally, do not provide any detailed information on when to train what to which level of detail using which methodology. Thus, a challenge lies in the design of appropriate training programs to support SPI initiatives that meet individual's training needs just before the knowledge is needed for work activities and that deliver the required knowledge cost-effectively with minimal schedule upheaval [9].

This paper relates our experiences in designing a tailored training program throughout a software process improvement initiative at the CYCLOPS Group (http://cyclops.telemedicina.ufsc.br) at the Federal University of Santa Catarina/Brazil. The CYCLOPS Group is a small R&D group that aims at the development and transfer of innovative methods, techniques and tools in the health care domain, including telemedicine, medical image analysis, 3D imaging, and workflow management in cooperation with several hospitals and medical clinics. Recognizing the need to improve its software process, the CYCLOPS Group started an improvement program in 2006. As part of the program, its software process has been organized and defined in accordance to CMMI-DEV [1], ISO/IEC 15504 [5] and MPS.BR [10] with a special focus on project management and requirements development and management.

2 Overview on the SPI Program

In 2006, we started the SPI initiative at the CYCLOPS Group adopting an enhanced version of ASPE-MSC (Approach for Software Process Establishment in Micro and Small Companies) [4], an approach aiming at a cost-efficient and effective improvement of software processes in small companies. As illustrated in Figure 1, the principal phases of the approach are Initiation, Diagnosis, Strategic Analysis, Definition and Institutionalization, which can be executed in an iterative and incremental way in order to improve step-by-step one or more process(es) within an organization. In addition, the approach also covers the management of the improvement of the software process(es), including planning, monitoring & control and post-mortem.

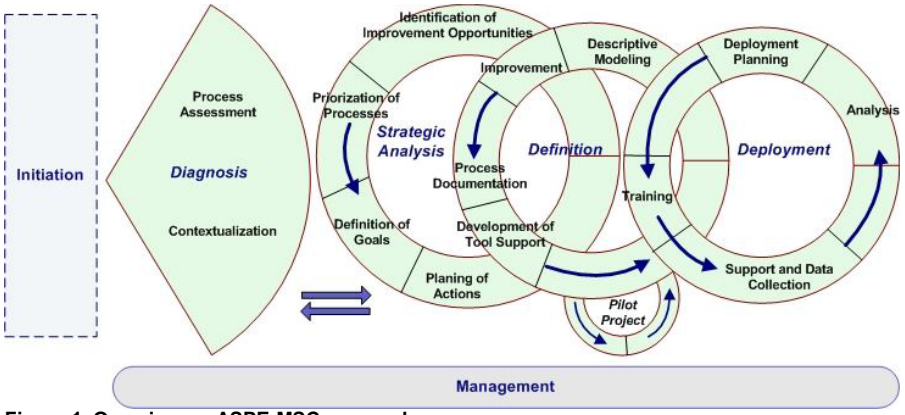


Figure 1. Overview on ASPE-MSC approach

During the initiation phase, a high-level planning of the SPI program was done and the required infrastructure was created. This included also the contraction of 2 process engineers (one full-time and one part-time) with competence in SPI as well as external consultants.

During the diagnosis phase, the context was characterized and the group's business and improvement goals were identified. We run a process assessment using MARES [3] in alignment with CMMI-DEV, ISO/IEC 15504 and MPS.BR providing as result the current process capability profile and a target profile. Based on this information, a strategic analysis took place, in which was decided to focus in a first improvement cycle on project management, in a second cycle on requirements development and management and in a third cycle on verification & validation. Based on this, the SPI program plan was refined. We, then, started to define, first, the project management process, covering project initiation, planning, monitoring & control and finalization.

Following the ASPE/MSC approach, the process definition was done by eliciting the actual process in place through process workshops and, then, based on a gap analysis improving the process in alignment with the reference models. The standard process and any related information to the SPI program was documented on the organizational WIKI. As part of the process definition, we also enhanced the open-source tool dotProject in conformity with the group's standard process. Once an initial version of the standard process became available, we introduced the process through pilot projects and periodically revised the process definition with the project managers. When the process definition became more stable, we started to institutionalize the standard process organization-wide. Required resources, information and infrastructure were made available and allocated. During the first weeks, the Software Process Improvement Group (SEPG) closely followed the process deployment and provided support. As part of the institutionalization, we also introduced periodic internal process audits in order to monitor and control the adherence of the defined processes. Feedback on the process deployment is used to revise and, if necessary, to adapt the standard process. In parallel to the institutionalization of the project management process, we started the second improvement cycle focusing on requirements development & management. Here we followed the same steps. Currently, we are continuing to monitor and control the process adherence with respect to the project management and requirements development & management process. Today, more than 70 percent of these processes are characterized at least as largely implemented in the majority of the software projects of the organization in relation to the standard process. So far, we have not yet initiated the third improvement cycle.

3 Training Program

Based on our experiences in performing SPI initiatives in small organizations or cooperative projects as well as other experiences related in literature [2, 7, 8, 11], we designed a comprehensive training program based on the actual needs throughout a SPI initiative. As a consequence, we realized various training sessions throughout the SPI program, with different learning objectives, audience and instructional methodology. Figure 2 illustrates the different training sessions performed.

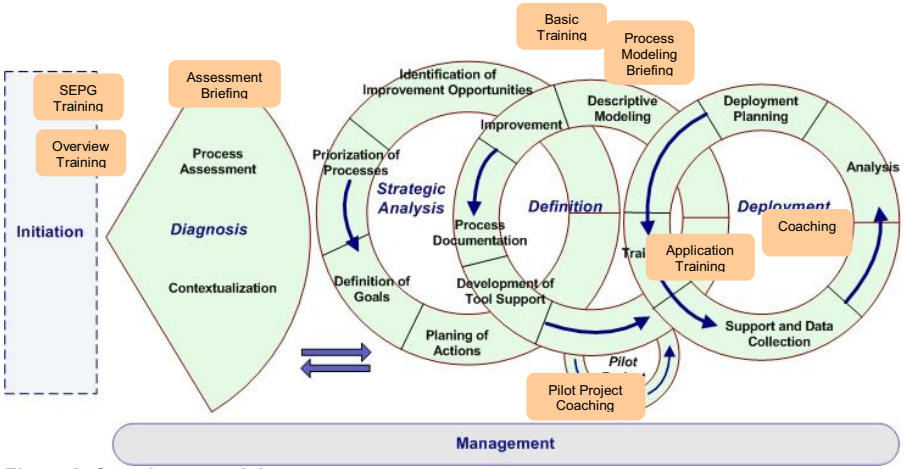


Figure 2. Overview on training program

SEPG Training. As a first step, we completed the training of the SEPG composed of two junior SPI consultants. Both SEPG members had already previous theoretical and practical knowledge on Software Process Improvement, reference models (principally CMMI and ISO/IEC 15504) and the process areas to be improved at the CYCLOPS group. In addition, both members of the SEPG participated in an official introductory course on the Brazilian process improvement model MPS.BR (16 hours). Further advanced topics in SPI were also covered in academic lectures, which both members attended as part of their participation in the Graduate Program on Computer Science at the Federal University of Santa Catarina. In addition, they were constantly assisted by the external consultants throughout the whole improvement program.

Overview Training. In the beginning of the improvement initiative, we also provided a one-hour overview training to all members of the CYCLOPS Group (including software analysts, programmers, managers etc.). The objective of the overview training was to provide a basic understanding on SPI, the improvement approach and reference models to be adopted. The training was also used in order to inform all members of the R&D group about the beginning improvement initiative, explain the steps and their planned involvement. One of the objectives was also to motivate the members and obtain their commitment. Therefore, the head of the group introduced the training and stressed the importance of the improvement program and the active participation of all members. The duration of the training was 1 hour and was held for all members of the organization. The training was provided by an external senior SPI consultant in form of an expositive lecture with discussion.

Assessment briefing. In the beginning of the process assessment, we provided a short assessment briefing in order to provide a basic understanding on the assessment objectives, the assessment process and the reference models to be considered. The objective was also to inform all assessment participants on when and how they would participate in the assessment and which information would be requested. Explaining the assessment objective and assuring the confidentiality of the information to be gathered was also intended to motivate them and obtain their commitment. The assessment briefing was provided in the beginning of the assessment itself and took about 30 minutes. The briefing was presented by an external senior SPI consultant in form of an expositive lecture with dialogue.

Basic training. Once a specific process had been selected for improvement, we carried out a basic training for this specific process. The objective was to provide a basic understanding on concepts, terminology and methods/techniques with respect to the process. The basic trainings were provided shortly before we started the elicitation of the respective process. The knowledge provided in these trainings is considered a pre-requisite for an effective definition of the process providing a common understanding and language to all participants. During the improvement cycles at the CYCLOPS Group, we realized the following basic trainings, so far as shown in Table 1.

Table 1. Overview on realized trainings

	Focus	Duration	Audience
1. Cycle	Project Planning	16 hours	Project manager
	Project Monitoring & Control	16 hours	Project manager
2. Cycle	Requirements Development	32 hours	System Analyst
	Requirements Management	08 hours	System Analyst

All basic trainings were provided by external senior SPI consultants in form of expositive lecture with discussion, including several concrete and practical examples and in-class exercises tailored to the specific context of the CYCLOPS Group.

Process workshop briefing. In the beginning of the process workshops, we provided a short briefing on process modeling presenting especially the graphical notation to be used. We also explained the objectives and expected usage of the process to be modeled in order to motivate the participants and obtain their commitment. The process workshop briefing took about 15 minutes and was presented to all participants of the process workshop, principally, representatives of performers of the respective process. The briefing was presented by an external senior SPI consultant in form of an expositive lecture with dialogue.

Pilot project coaching. In order to enable participants of pilot projects to successfully execute the process, we provided on-the-job coaching involving actively the SEPG within the pilot projects (ranging from 2 – 8 hours weekly). No formal training was provided at this moment of the SPI initiative, as the standard process was still being defined and, as the number of people to be trained was limited to

the personnel of the pilot projects. In addition, the intensive involvement of the SEPG assured a constant feedback and discussion of critical aspects of the standard process and facilitated also its evaluation.

Application training. Once an organizational standard process (e.g., on project management) had been defined and was being institutionalized, we provided an organization-wide training session on the standard process, presenting its objective, its execution step-by-step as well as demonstrating tools to be used during the process execution. The objective of the application training was to inform all process performers about the standard process and enable them to execute the process as defined. The application trainings with a duration of 2 hours each were presented by the SEPG of the CYCLOPS Group. We used various instructional methods, including an expositive lecture with an execution example based on a real case from the CYCLOPS Group. As part of the example, we also demonstrate the tool usage during the process execution. In order to further illustrate the standard process and to emphasize changes, we used two custom produced videos: one showing an exemplar execution of the informal process used before and one showing the newly defined standard process. The videos were set in an actual laboratory of the R&D group, starring members of the CYCLOPS group as actors. In order to assure the participation of basically all members of the CYCLOPS group, we offered identical application trainings at two different times, immediately before we started the organization-wide deployment of the respective process.

Coaching. Throughout the institutionalization of the processes, we are providing training in form of coaching. Coaching takes place periodically together with internal process audits, which have been established in order to monitor & control the deployment of the processes. During these audits, the SEPG assesses the adherence of a project's execution in relation to the organizational standard process. As a result of the assessment, detailed to-do lists on what has to be improved are provided as well as a general evaluation of the degree of adherence. Coaching then takes place by explaining the actions to be taken, assisting the process performer on-the-job in order to guide and to facilitate the process execution and by discussing any problems with regard to the standard process. In the beginning of each process institutionalization, we spent a considerable amount of effort on coaching summing up to approx. 20 person-hours/week by the SEPG. However, after the initial phase, we are continuously reducing this effort to now about 2 person-hours/week. Coaching also takes place on demand, when a process performer has any question or difficulty with respect to the processes being improved.

4 Lessons Learned

Comparing our experiences in designing the training program for software process improvement at the CYLOPS Group to other experiences in other SPI initiatives, we observed several lessons learned based on evaluations of the training sessions themselves as well as subjectively perceived learning effects:

Training tailored to the specific context. One of the principal strengths of this training program was its tailoring to the specific context of the CYCLOPS Group. We customized all training sessions, using concrete situations and examples of the process execution and work products from the organization, which we elicited in cooperation with process performer. Comparing such a tailoring to more generic of-the-shelf trainings, as, e.g., typically applied in cooperative SPI programs, we perceived a significant difference on the understanding as well as the motivation, as tailored trainings illustrate much more convincingly which problems are to be solved and how. Tailoring training courses to a specific context, of course, requires a larger amount of preparation, yet the achieved benefits seem to justify the higher costs. In addition, the trainings were all provided in a form, which stimulated discussion and the exchange of experience of members, which in itself directly resulted in some improvements, besides the learning and engagement effect with respect to the training.

Only as much and when necessary. Especially compared to trainings typically offered as general training sessions in blocks, we considered the subdivision of the SPI training in several sessions at different moments of the SPI program a substantial strength. By dividing the training into several training sessions we were able to focus on relevant competence required at a particular step of the SPI initiative. As a consequence we were able to provide tailored training for specific audiences keeping

focus on exactly the knowledge, which was important for them. This had various positive outcomes: increased learning effect as exactly the needed knowledge was taught and reduced time spent in training, as the sessions were more focused and only people, which were in fact involved, participated. We also observed that the timely provision of the trainings, which allowed the participants to apply the knowledge immediately, afterwards, further motivated and increased the learning effectiveness.

Training can be fun. As one of the principal strengths of the trainings, the members of the CYCLOPS group cited the videos presented during the application training. We experienced the usage of such custom made videos, showing the standard process in a way the audience can directly relate to and even enjoy, an opportunity to receive the audience's full attention. Especially the fact, that the videos were produced within the CYCLOPS laboratory presenting real life scenes with actual members of the group increased the engagement during the whole training. An additional advantage of such video material is also it's just-in-time availability that accommodates individual schedules and needs as well as the training of new members and its aid to retention due to its ability to be viewed repeatedly, especially when filed with additional training material and process guides. We consider the production of such short videos (both of about 5 minutes duration) rather simple and worth the additional effort. The videos (figure 3) were produced in an inexpensive way by the SEPG in cooperation with a journalism student of the university. As a basis a high-level script was sketched and then filmed without any further preparation. The total effort for the production of the videos was about 40 person-hours.

Figure 3. Training videos

Training in cooperation with SEPG. For the training program at the CYCLOPS Group, we customized general training material based on information provided by process performers. This contact with the organizations' members also helped to create an open atmosphere characterized through mutual respect. The trainings were prepared and provided in cooperation by the external consultants and SEPG, coaching also the SEPG on-the-job in order to enable them to assume this responsibility in future improvement cycles.

Constant coaching during deployment. Based on our experience, one of the most important factors for the successful institutionalization of a standard process are its monitoring & control (e.g., through internal process audits) and constant coaching throughout the organization-wide deployment. We observed that, although, the application training was evaluated as adequate for understanding the process execution, process performers encountered various problems and questions, when actually starting to execute the process or process were simply not executed in alignment with the standard process as identified through audits. In this situation, we started to prepare concrete to-do lists based on the audit results and provided on-the-job coaching in which SEPG members assisted the project performers closely throughout the first process executions, if necessary. With progressed process institutionalization, we are now reducing this coaching effort, but maintaining a minimum in order to keep a communication channel with the process performers to obtain feedback on the process application.

5 Conclusions

Based on the comments and evaluations of the participants of the trainings and the subjectively perceived learning effect in practice, we consider the training program at the CYCLOPS group a success. Another indication is also the fact that various members of the group requested further trainings on subjects related to software process improvement, which shows the motivation and interest in SPI they developed. This one application is, of course, not sufficient to generalize the obtained results, yet, our approach may also serve as an example for training programs in other SPI initiatives. We intend to repeat the proposed training structure in other programs and perform more formal evaluations of the trainings and the achieved learning effects.

Acknowledgements

Our thanks to all involved in the software process improvement initiative at the CYCLOPS group/UFSC.

Literature

[1] CMMI Product Team. CMMI for Development (CMMI-DEV), Version 1.2. Technical Report CMU/SEI-2006-TR-008, Carnegie Mellon University/ Software Engineering Institute, Pittsburgh, August 2006.

[2] P. Fowler, S. Rifkin. Software Engineering Process Group Guide. Technical Report CMU/SEI-90-TR-024, September 1990.

[3] C. Gresse von Wangenheim, A. Anacleto, C. Salviano. Helping Small Companies Assess Software Processes. IEEE Software, Vol. 23, No. 1, Jan/Feb 2006.

[4] C. Gresse von Wangenheim, S. Weber, J. C. R. Hauck, G. Trentin. Experiences on Establishing Software Processes in Small Companies. Information and Software Technology, v. 48, n. 9, 2006.

[5] ISO/IEC Std. 15504: Information Technology – Process Assessment, Part 1 to Part 5. International Organization for Standardization, 1998-2006.

[6] C. S. McCahon, M. J. Rys, K. H. Ward. The impact of training technique on the difficulty of quality improvement problem solving. Industrial Management & Data Systems, Vol. 96 No. 7, 1996.

[7] R. McFeeley. IDEAL: A User's Guide for Software Process Improvement. Handbook CMU/SEI-96-HB-001, Software Engineering Institute/Carnegie Mellon University, Pittsburgh, 1996.

[8] P. O'Toole. Do's and Don'ts of Process Improvement, SEPG Conference, March, 2004.

[9] R.S. Pressman. Software Process Impediment. IEEE Software, Sep. 1996.

[10] SOFTEX. Brazilian Software Process Improvement Model MPS.BR. (http://www.softex.br/mpsbr)

[11] K. E. Wiegers. Software Process Improvement: Ten Traps to Avoid. Software Development, May 1996.

Author CVs

Christiane Gresse von Wangenheim

Christiane Gresse von Wangenheim is a professor at the Universidade do Vale do Itajaí (UNIVALI) and consultant at Incremental Tecnologia Ltda. Her research interests are software process improvement, including project management. Previously, she worked at the Fraunhofer Institute for Experimental Software Engineering. She received a PhD in Production Engi-

neering at the Federal University of Santa Catarina (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. She's a member of the IEEE Computer Society, the Project Management Institute, and the Working Group ISO/IEC JTC1/SC7/WG24—SE Life-Cycle Profiles for Very Small Enterprises. Contact her at UNIVALI, Rod. SC 407, Km 04, 88122-000 São José/SC, Brazil; gresse@gmail.com

Jean Carlo R. Hauck

Jean Carlo Rossa Hauck is SEPG manager of the CYCLOPS Research Group at the Federal University of Santa Catarina (UFSC). His research interests are in software process improvement and project management. He received his M.Sc. in Computer Science from the Universidade Federal de Santa Catarina and is a PhD student of the Graduate Program in Knowledge Engineering and Management at the Federal University of Santa Catarina. Contact him at UFSC - EGC, Campus Universitário 88049-200 Florianópolis/SC, Brazil; jean-hauck@egc.ufsc.br

Richard H. de Souza

Richard H. de Souza is a master student of the Graduate Program in Computer Science at the Federal University of Santa Catarina. His research interests are in software process improvement and requirement management. He received his B.Sc. in Computer Science from the Universidade do Vale do Itajaí (UNIVALI). Contact him at UFSC -CTC-INE, Campus Universitário 88049-200 Florianópolis/SC, Brazil; richardhenrique@gmail.com

Maiara Heil Cancian

Maiara Heil Cancian is system analyst of the CYCLOPS Research Group at the Federal University of Santa Catarina (UFSC). Her research interests are software process improvement and project management. She received his B.Sc. in Computer Science from the Universidade do Vale do Itajaí (UNIVALI) and is a master student of the Graduate Program in Automation and Systems at the Federal University of Santa Catarina. Contact her at UFSC -CTC-DAS, Campus Universitário 88049-200 Florianópolis/SC, Brazil; maiara@telemedicina.ufsc.br

Networks and Learning Organisations – Innovation and Team Learning Supports Improvement Programs

Richard MESSNARZ[1], Gunther Spork[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
[2]*Magna Powertrain,Lannach, Austria*
Email: bruno.woeran@danube.or.at

Abstract: The paper will illustrate some example results of an EU project ORGANIC – where learning strategies for organisations have been developed, trialed in companies, and delivered in form of certified training.

The original project was called **ORGANIC** (2005 – 2007) by purpose. Based on different European studies about innovation management where members of the partnership and leading industry have been involved we developed a modern learning organisation based innovation management strategy. A company becomes an ORGANISM where through continuous learning spirals the knowledge grows and the core competences increase continuously. In collaboration with innovation leading companies the project developed an example base which is being exchanged and used in different working task forces since 2006. This is also reflected in the way the innovation manager is transported to industry.

In this paper we want to emphasise that learning strategies and a structured approach to turn organisations into learning organisms are a major influence on the success of improvement programs.

Meanwhile the European Union finances a project called EU Certificates campus (2008 – 2010) where such key areas of knowledge are transported in form of online short courses, together with recognised certificates for innovation management.

1. The modelling of a learning strategy

In ORGANIC [9] we run through 20 competence areas when organising a firm into a learning organisation:

- Building Basic Understanding
  - **Core Competencies and Customer Relationship Management Skills**
  - Innovation and EU Policies Know-how
  - Introducing Innovation Management Principles
  - Knowledge Management Competencies
  - Market Research Skills
  - Regional Innovation Strategies Involvement
  - Human Force Skills Management
- Building Communication Skills

    - e-Challenges in Innovation
    - Innovation Skills for Reporting&Presentation Skills
- Building Management Skills
    - Corporate Wide Innovation Management
    - Innovation Aspects in Project Management
    - **Innovation Process Management Process**
- Building Team-Learning and Teamworking
    - Cross Cultural Success Factors
    - Innovation Aspects in Conflict Management
    - Innovation Aspects in Motivation Building
    - Innovation Aspects in Team Communication
    - **Innovation Skills for Distributed Team Management**
- Building Personal Skills
    - Cross Cultural Skills
    - Knowledge about Personal Characteristics
    - Learning Culture Establishment

Each of the competence areas has equal importance. For each of the competence areas the material proposes certain best practices and asks the participant to apply that on the own organisation.

After running through all steps you have created the architectural design of a learning organisation tailored to your own needs.

We highlighted three areas in the above listing because we will explain these with examples in the following sections of the paper.

2. What is a learning organisation?

A learning organisation [5],[6],[8],[9] creates a positive learning culture and enables team learning and synergy exploitation in an organisation. By team learning knowledge is spread much more quickly and a high level of a skilled human force is maintained.

Typical examples of failure are

- You recognise that for the implementation of a new product or new processes you lack specific skills and have no chance of acquiring them in time.
- You recognise that departments inside the company have the knowledge but do not want to share it with other departments.
- You recognise that your competitors have formed a group to share knowledge and jointly compete against you on the market.
- You recognise that some of your management staff does not fully understand the mission.
- You recognise that someone in your firm bought a knowledge management system but none uses it.
- Etc.

Typical examples of success are

- You linked in time yourself to experience partnerships and training networks and can react on the market immediately with any skills required.
- You manage that knowledge and team learning is used in a synergy approach between the departments and teams.
- You were the one who formed the group that jointly learns and shares knowledge and collaborates against your competitors.
- You ensure that the mission is a goal which binds everyone to a big picture.
- You analyse the core knowledge (the one that differentiates you from the competitors) and build all knowledge management strategies around that core (=realistic and not holistic knowledge management!).

- Etc.

In learning organisations there is an infrastructure in place which enables the team learning and the spreading of knowledge and team communication.

3. Samples of Implementation

The full work is published at www.innovationmanager.org and certificates are issued by www.eu-certificates.org, with iSQI (International Software Quality Institute) as certification body.

3.1 Core Competence Analysis and CRM Example

One of the key success principles is that organisations understand that they are part of a learning chain. The innovation ideas of customers influence their own innovation tracks. The closer one gets to such key partners the more dynamic the learning cycles will flow.

Step 1. Identify Core Competence

A core competence is a field of knowledge of the firm
- Were they are stronger than other competitors.
- Where they created already a critical mass of competence.
- Where with one knowledge item / function many customers can be served (Re-usability).
- Where since years dynamically knowledge is extended, newly created and exploited.

Step 2. Identify Key Customers for Learning

Once an organisation identified the core competence fields the next step is to identify which customers are those who most dynamically contribute ideas to this core competence.

A key learning customer is identified as a firm which
- Regularly gives inputs to new functions, ideas, plans for increasing the identified core competence
- Has its own known innovation leadership and can help putting new structures into place
- Is willing to get in closer collaborative partnerships for services and products in the future

Step 3. Enable a Social Learning Strategy

Once the key customer and the core competence are identified the organisation creates supportive social learning spaces to further enrich the communication and empower the dynamic feedback flow to the core competence.

In **Figure 1** we illustrate the example of Automotive Systems [7] where core functions of e.g. a control system are the same in all variant projects. The company then decides to develop all base functions just once and maintain parameter sets which allow to apply the same 80% (ready-to-use) functionality by parameter sets to many different customers. The company then learns continuously new functions and decides whether to include them in the base.
This leads in the long run to stable systems working for many customers and focussing the learning on core functions which they can supply better and quicker than any of the competitors.

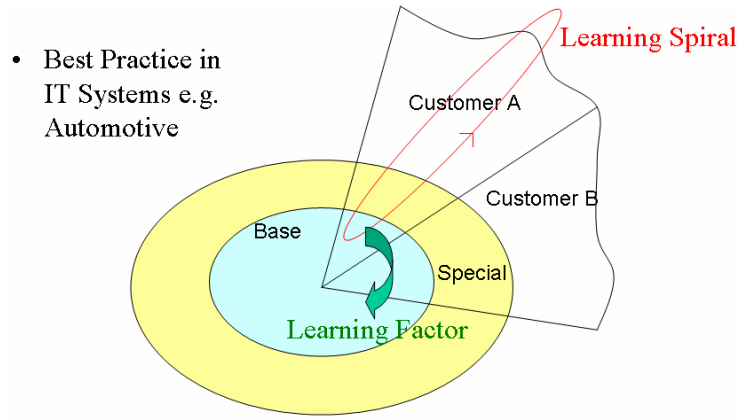


Figure 1: Example – Base Development Strategy in Automotive Systems Development

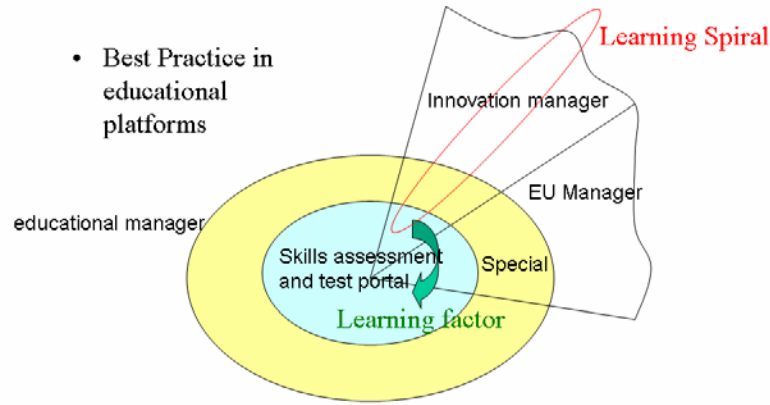


Figure 2: Example – Core Content Strategy in European Skills Portals Development

In **Figure 2** we illustrate the example of a European strategy (EU Certificates, PLATO, MMPS) where a base system with sills set structures and functionality to support skills assessments and exams for European professions is developed one time and adapted to many professions.
The learning spiral is then driven by the professions which deliver most ideas how to further increase the base functionality of the system.

In **Figure 3** we illustrate the example of a leading automotive supplier which identified using the analysis that e.g. Customer B would drive the innovation in the mechatronics functions while currently the base knowledge for safety design is created with the idea motor Customer A.
Based on the specific team structures are built to further increase this learning spiral.

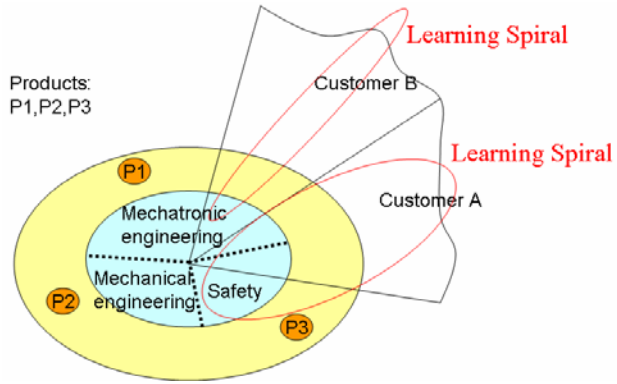


Figure 3: Example – Core Knowledge Strategy in Competitive Development

Benefits

Imagine that you either do 30 parallel projects (30 times the effort, 30 different results, 30 maintenance teams, etc.) or that you do one core competence team that provides one solution adapted (by parameters and configuration options) to 30 variants of customers.
You can focus knowledge, you can focus resources, and you can focus on customers that are contributing further to the core knowledge.

Projects then start with 80% ready functions and you overtake competitors by a timing of 1:5.

Relationship to SPICE

Imagine that you do 30 projects and have to create a tailored process, requirements tree, test plan, etc. for 30 projects. Or that you do this for one core project that contributes to 30 variant projects. You invest one time and it pays back 30 times.

3.2 Innovation Skills for Distributed Teams Example

Another key success principle is that organisations are able to model and support the learning spiral (see the learning spiral in Figures 1,2,3) in form of a role based distributed team [1], [2], [3] [4] . This way they learn a so called learning cooperation pattern which can be re-used to dynamically run these learning / innovation partnerships.

Let us continue with the example in Figure 3 and how the core competence for safety design further developed. The company then analyses what are the currently involved roles and the current information flows in that safety related learning cycle.

In **Figure 4** we illustrate the current levels of roles involved in the safety concept, safety design and safety implementation. In **Figure 5** we illustrate the current information flow which showed that there is a bottleneck with the safety manager.

The results of such an analysis would be shown in **Figure 6** where the learning organisation would decide to create a joint learning time to unleash the power of knowledge exchange and collaboration.

Step 4. Analyse Current Team Roles

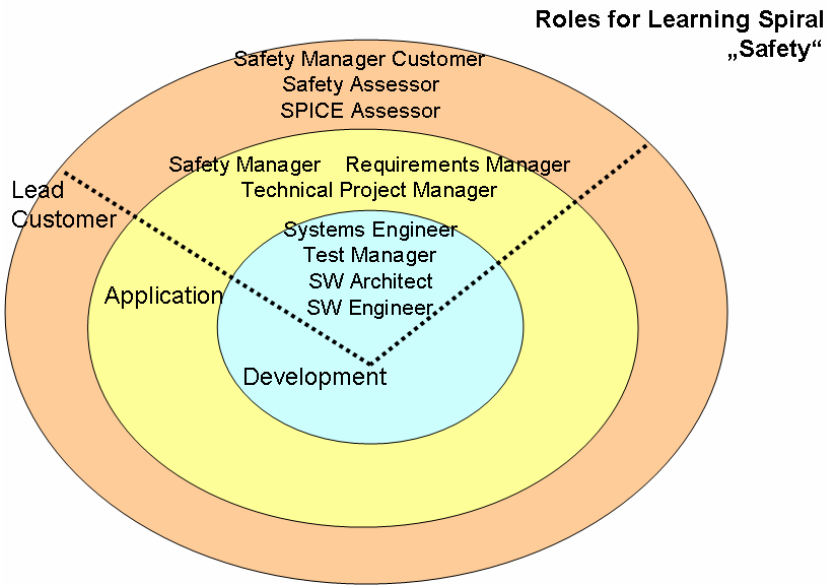


Figure 4: Example – Safety Learning Cycle in Figure 3 –Actual Roles

Step 5. Analyse Current Team Flows

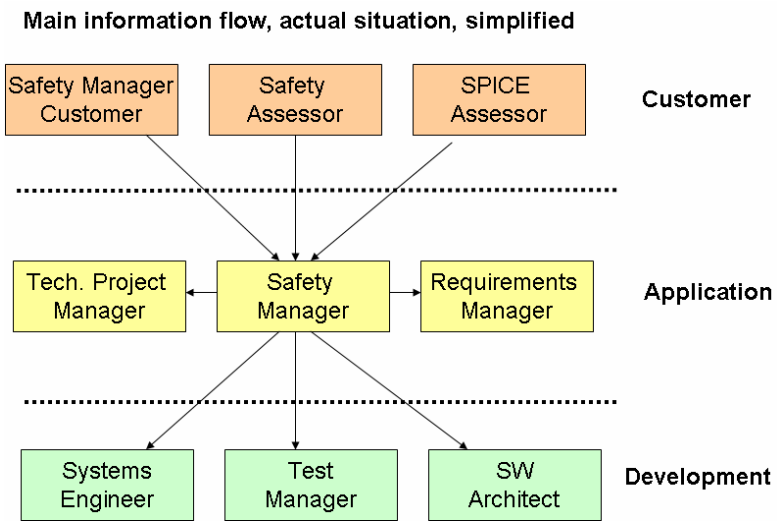


Figure 5: Example – Safety Learning Cycle in Figure 3 –Actual Flows

Step 6. Improve towards a learning team

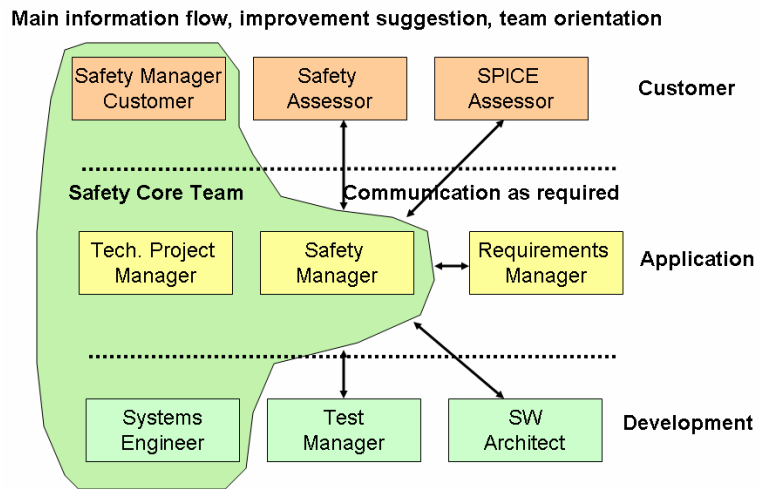


Figure 6: Example – Safety Learning Cycle in Figure 3 – Social Team Learning

Distributed Innovation / Learning Teams

A distributed innovation / learning team

- Involves roles from different levels (customer, product, core competence)
- Does not have bottlenecks
- Enables teamwork and feedback loops to create ideas, solutions, knowledge
- Distributes and shares information to the team members

Benefits

The learning effect on the core knowledge (safety design in that example) is multiplied by bringing key players together in a learning team. Much information and time is lost when bottlenecks serve in the middle. Also, remember, we need to further increase the dynamics around the learning cycle and the faster it turns the more we learn together on e.g. safety design.

Projects then have access to a quicker generated knowledge base which helps them to re-use that in all variants and further projects.

Relationship to SPICE

Imagine that in level 2 assessments assessors always ask about generic practices 2.1.4 to 2.1.6 which relate to team roles and performance. In this case we ask for optimised team structures involving customers, engineering and key staff on e.g. important safety decisions.

3.3 Innovation Process Management Example

In most traditional innovation management courses the content relates to patents, supporting new patents, creating idea databases and following up on the ideas, supporting innovative staff, etc.

In learning organisations we add to this traditional picture the organisational strategy of a continuous learning organism around features which keep the organisation alive and leading for a long time.

Therefore another key success principle of learning organisations is the ability to create innovation processes around the learning dynamics of the organisation [1], [3], [7], [9].

Feedback Loop Based Innovation / Learning Processes

Feedback loop based innovation / learning processes
- Must represent continuous feedback loops
- Are created based on the learning cycles
- Support the continuous increase of core competence knowledge
- Create critical mass of knowledge to be re-usable in many projects and services

Step 7. Create an Innovation Process based on the Learning Cycles

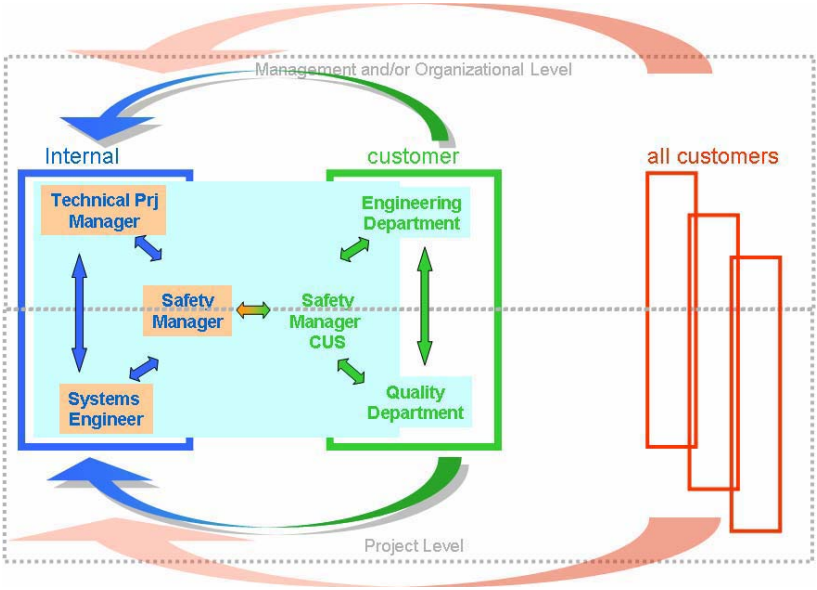


Figure 7: Example – Safety Learning Cycle in Figure 3 – Feedback Loop Processes

In **Figure 7** we illustrate a picture of a designed feedback loop process around the safety core team. Customer and project roles collaborate closely, gather key knowledge prepared and stored by the internal team, and continuously refining the knowledge based on planned feedback loops.

Benefits

Imagine that you do many projects and each contributes core knowledge and you safe it just in the project space. Then the knowledge will stay in each single project and eventually (if a staff member moves to another project) be shared.

Or imagine that you declared certain knowledge as core knowledge, projects share together, and a base structure (product, service, knowledge, requirements tree, etc.) is built for all projects in the centre. Then all knowledge flows together and the feedback loop process is a strategic process in the firm.

Relationship to SPICE

Imagine that e.g. (to continue with the safety design example) all projects developing a similar function use a different safety requirements tree (with links to tests). Then you must do the same audit and tests many times.
Imagine that you created a safety component used in a group of projects, collect all knowledge in that, then you need one audit, one safety concept with little variations, and one set of tests to be repeated.

4. The Grand Strategy

Learning Organisation Related

The framework for designing a learning organisation has 20 competence areas [9] . Each area has its own success principles. By running through all 20 areas an architectural design for a learning organisation is created.

The training and certificate to learn about these 20 areas (also about how to implement the principles) is being called the “EU Certified Innovation Manager”. The certificate is issued by iSQI (International SW Quality Institute).

ISO 15504 / SPICE Related [10]

Knowledge about these innovation principles is important for SPICE assessors to provide improvement recommendations which help organisations to win from SPICE investments.

- To know the core competence in functionality in a product segment will help to install the requirements and test traceability for a core functionality once, and then repeat to use it from there. So the investment pays back many times.
- If the understanding of customer, system, and software requirements is demanded, then such learning teams are the basis for such a good communication.
- Innovation is based ion a continuous learning cycle involving the customer and core competencies which can be multiplied into many product segments and projects.

Outlook

Cross company learning teams on core areas of SPICE have been created in SOQRATES 2003 (www.soqrates.de), where up to now above 20 German leading firms collaborate.

In **Figure 8** we illustrate a picture of the collaborative innovation learning model applied in SOQRATES. Clusters of companies are formed who can contribute key knowledge to a SPICE core competence. Companies can only join on a win-win principle where they give (be a key player to one of the knowledge fields) and take (can access core knowledge elaborated by another cluster team).

Still it is exclusive to be a member of the group because existing members must agree the integration of new members. Thus the core group contributors are no competitors, they exchange and learn from each other, and get together better than their competitors on the market.

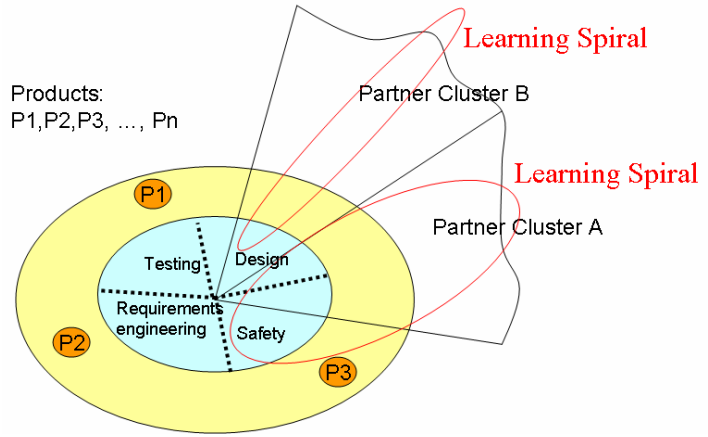


Figure 8: Example – Core Competencies Architecture for Cross Company Task Forces Model

Using the same innovation learning strategy cross company learning teams on core areas of SPICE have been created also in Austria in S^2QI 2005, where up to now above 10 leading Austrian firms collaborate.

In 2005 the innovation manager consortium was founded (www.innovationmanager.org) which is now continued to be supported by the EU in EU Cert (2008 – 2010, www.eu-certificates.org).

If you plan to participate in a training partnership then contact iSQI (www.isqi.org).

References

[1] M. Biro, R. Messnarz, A. Davison (2002) The Impact of National Cultures on the Effectiveness of Improvement methods - The Third Dimension, in Software Quality Professional, Volume Four, Issue Four, American Society for Quality, Sep-tember 2002

[2] Feuer E., Messnarz R., Wittenbrink H., Experiences With Managing Social Patterns in Defined Distributed Working Processes, in: Proceedings of the EuroSPI 2003 Conference, 10-12 December 2003, FTI Verlag, ISBN 3-901351-84-1

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

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

[5] 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

[6] R. Messnarz, et. al, Assessment Based Learning centers, in : Proceedings of the EuroSPI 2006 Conference, Joensuu, Finland, Oct 2006, also published in Wiley Interscience Journal, SPIP Proceeding in June 2007

[7] G. Spork, et. al, Establishment of a Performance Driven Improvement Program, in : Proceedings of the EuroSPI 2007 Conference, Potsdam, Germany, Sept. 2007, also published in Wiley Interscience Journal, SPIP Proceeding in June 2008
[8] R. Messnarz, et. al, Human Resources Based Improvement Strategies – the Learning Factor, in : Proceedings of the EuroSPI 2007 Conference, Potsdam, Germany, Sept. 2007, also published in Wiley Interscience Journal, SPIP Proceeding in June 2008
[9] Messnarz R., et. al., ORGANIC - Continuous Organisational Learning in Innovation and Companies, in: Proceedings of the E2005 Conference, E-Work and E-commerce, Novel solutions for a global networked economy, eds. Brian Stanford Smith, Enrica Chiozza, IOS Press, Amsterdam, Berlin, Oxford, Tokyo, Washington, 2004
[10] ISO / IEC 15504 Standard, Parts 1-5

ORGANIC - Continuous Organisational Learning in Innovation and 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).

Dipl. Ing. Gunther Spork

Gunther Spork has a Master of Science degree in mechanical engineering from the Technical University at Graz.

He has a long experience in a wide area of improvement. After starting as manufacturing engineer for VA TECH Hydro he joint a Six Sigma Team which was responsible for deploying a Six Sigma pilot structure in 1998. Beside being in charge for the Six Sigma project success in the Generator Division as Master Black Belt he was also responsible for the Six Sigma training program and trained colleagues from 3 continents. Additionally he participated in strategic projects of the VA TECH Group.

In 2001 he started to work for Magna Powertrain in Lannach and was responsible for the Management System according ISO TS 16949 and Six Sigma. Later he took over the responsiblility for engineering processes and development methods. In this function he is also responsible for the deployment of SPICE. The aim for improvement goes along with

internal and external ISO 15504 assessments according automotive SPICE regulations.

Dipl. Ing. Damjan Ekert

Dipl. Ing. Damjan Ekert is the chief developer of the Capability Adviser and EPI / Learning systems since 2003. He studied Telematics in Austria and finished studies with distinction. He is a certified ISO 15504 assessor and works in consulting projects for Magna. He is the project leader for software development inside ISCN.

Building a theoretically reflective model of software engineers' motivators

Nathan Baddoo[1], Sarah Beecham[1], Tracy Hall[1], Hugh Robinson[2], Helen Sharp[2]

[1]School of Computer Science, University of Hertfordshire, College Lane, Hatfield, Herts AL10 9AB, UK
[2]Dept of Computing, Faculty of Mathematics and Computing, The Open University, Walton Hall, MK7 6AA, UK
({n.baddoo; s.beecham; t.hall; }@herts.ac.uk) ({h.m.robinson; h.c.sharp; }@open.ac.uk)

Abstract

We present a model of motivation for software engineers. Our model suggests that software engineers are motivated by two sets of factors, intrinsic and extrinsic motivators, where a subset of intrinsic motivators are aspects inherent in the job that software engineers do. It shows that software engineers are orientated towards these particular sets of motivators because of their characteristics, which in turn are mediated by individual personality traits and environmental factors. Our model shows that the external outcomes of software engineers' motivation are benefits like staff retention, increased productivity and reduced absenteeism. Our model is derived from a Systematic Literature Review of motivation in software engineering. We have constructed this model by engaging in practices that reflect good principles of model building as prescribed by operational research and scientific management discipline. We evaluate our model for theoretical efficacy and show that our model, in comparison to other attempts at modeling software engineers' motivation, reflects a wide range of the classic concepts that underpin the subject area of motivation. We argue that, this theoretical efficacy validates the model and therefore improves confidence in its use. We suggest that our model serves as a valuable starting point for managers wanting to understand how to get the best out of software engineers, and individuals wanting to understand their own motivation or who are embarking on career choice.

Keywords: motivation, model, theory, software engineer, systematic literature review

1. Introduction

In this paper we present a model of motivation for software engineers. We derive this model from a Systematic Literature Review (SLR) of software engineers' motivators. We follow model-building principles from the discipline of operational research and scientific management to construct this model and evaluate the model in terms of how it reflects classic theories of motivation. We show that our model is strongly corroborated by existing phenomena in the organizational psychology literature on motivation.

Motivation is increasingly cited as a particularly pernicious people problem in software engineering. In DeMarco and Lister's 1999 survey motivation was found to be one of the most frequently cited causes of software development project failure [DeMarco & Lister 1999]. The Standish report [1994] amplifies this finding by reporting that having access to competent, hard working and focused staff is one of 10 success criteria for software projects. However, until now no comprehensive model of what motivates software engineers has been developed. Consequently it is difficult for managers to know how best to motivate their software engineers.

Our model of motivation is based on findings from a systematic literature review of 92 published papers. This means it is rigorously underpinned by previous work in the area. Our model shows that there is a complex array of factors that must be managed effectively to get the best outcomes from software engineers. Our model also reflects some classic concepts of motivation, which is an important outcome of this work, since our related work suggests that theories are not used well in existing studies of motivation of software engineers [Hall et al 2007].

We present our model in terms of how it reflects classic motivation theory because we believe that it is important to support the representation of existing phenomena with the theories that give credence to those phenomena. For example, we argue that even though the Wright brothers initiated the modern concept of sustainable flight, the

classic and underpinning phenomenon of flying has existed for as far back as antiquity, evidenced by the fatal flight of Icarus to the Sun. Our underlying argument is that most concepts and theories of human existence and behaviour already exist in a classic form. However, it is their realization in contemporary forms and contexts that tend to give us more and better insights into these concepts. Therefore in our endeavour to study the dynamics of software engineers' motivation, we want to ensure that any model of motivation that we derive through empirical observation, can be corroborated by some existing and classic concepts of that subject. This is why in addition to discussing the model construction process, we also focus on how the model that we present here reflects classic motivation theory.

1.1 The importance of modeling motivation

Understanding the human factors in software engineering is well known to be critical to the success of software projects [Tomayko and Hazzan 2004]. Much research on human factors has concentrated on user issues rather than developer issues. However, increasingly it is being recognised that the human factors associated with developers themselves are critically important to good software engineering outcomes [Sharp and Robinson 2005]. Motivation is one such developer issue that is poorly understood yet has been cited as important to project outcomes. According to Brooks [2005] even when the focus is right much of the progress in the area of motivating software engineers to work better has tended to concentrate on the reduction of environmental factors that impede their productivity, and not so much on what improves performance - the motivators.

In this respect, we identified two gaps in the area of research on software engineers' motivators. Firstly, work that has been done over the last 30 years on the motivation of software engineers has not been analysed in a systematic way. Our systematic literature review therefore pulls together this previous work to show the landscape of work done in the area. This allows us to understand the expanse of work in the area and identify how the motivation of developers has changed over time as the software engineering environment has evolved.

Secondly, that despite the volume of work done on motivation generally, there has been insufficient effort invested in the simulation and modeling of software engineers' motivation. Models help to illuminate our understanding of concepts and practices [Pidd 1996], so insufficient investment in this area of software engineering must be frustrating our efforts to understand software engineers' motivators.

Our SLR uncovers a variety of previous attempts at developing models of motivation in software engineering. Our findings show that although there is valuable work in the area, the models that have been proposed are fairly disparate and insubstantially underpinned. As a result, no concrete model exists that attempts to show how software engineers are motivated.

We therefore embarked on a process of building a model of software engineers' motivators from the results of our SLR, and aided by the precepts and principles of model building from the discipline of operational management. Additionally, because Systematic Literature Reviews provide a source of secondary data [Kitchenham 2004], we used the findings from our SLR as provisional data to support the construction of our model by populating the contents of the model structure with these findings. Later refinements of this model can be found in Sharp et al [2007] where findings from our SLR of other models of motivation of software engineers are used to derive a second cut model. In this paper, we concentrate on the construction of the first cut model and show how we derived the second from the first.

Our model represents motivation as a series of stages in a process. It suggest that software engineers are motivated by two sets of factors, intrinsic and extrinsic motivators, where a subset of intrinsic motivators are factors that are aspects inherent in the job that software engineers do, for example, the problem solving nature of software engineering. Our model suggests also that the factors that orient software engineers toward these particular motivators are his characteristics and that these characteristics are mediated by individual personality traits and environmental factors. Our model shows that the external outcomes of a software engineer's motivation are staff retention, increased productivity and reduced absenteeism. Overall, we show that both the dimensions and dynamics represented by this model of software engineers' motivation are reflected in classic motivation theory.

We suggest that our model serves as a valuable starting point for managers wanting to understand how to get the best out of software engineers, and individuals wanting to understand their own motivation, or who are embarking

on career choice. Our model also provides a platform from which subsequent researchers can base their empirical studies.

In Section 2, we provide some background on the value of models. We also present an overview of motivation theories. In Section 3, we describe our research methods and in particular our systematic literature review protocol. In Section 4, we present the findings of our SLR, detailing how these findings underpin the development of our model. In Section 5, we present our model and evaluate it in terms of how it reflects classic and conventional concepts of motivation in Section 6. We conclude and discuss our future work in Section 7.

2 The value of models

In this section we discuss the concept of models, present some suggested principles on model development and the importance of ensuring that any model developed sufficiently reflects some underlying theory.
We argue that any model derived to examine a particular area of study or practice should reflect the classic theories in that area. Indeed, work that we have done shows that some models are actually representations of theory. We provide a summary of theories of motivation. An extensive overview of these theories is published by Hall et al [2007]. We will, in later sections, reflect the model presented in this paper in the light of these theories.

2.1 Why are models important?

Pidd [1999] defines a model as:

> "An external and explicit representation of part of reality as seen by the people who wish to use the model to understand, change, manage and control that reality in some way"

The above definition presents a model as external and explicit, as opposed to a notion or concept represented by mental constructs [Pidd 1999]. Pidd also stresses that no model will be a complete representation of reality, since such a model will be too complicated, expensive and hard to manipulate. Instead, models tend to make partial representation of the real world and such partial representations should be fit for some purposes, and not necessarily all purposes [Pidd 1999]. In fact, Jenkins and Youle [1976] and later Pidd [1999] describe two types of model building approach where one is aimed at trying to model the complete reality, an approach favoured by chemists and physicists [and the pioneers of Software Systems Methodology (SSM), incidentally] and the other that attempts to use the model as an intermediary step to understanding the full intricacies of the concept being modeled [Jenkins and Youle 1976]. Our approach to model building in this work follows the intermediary step suggested by Jenkins and Youle. We do, however, appreciate that some model builders in software engineering follow the classicist approach described earlier.

Models are used in all fields of software engineering, from requirements engineering to software evolution. Throughout the spectrum of software engineering, models have been used in the classic way that operational managers and management scientist describe model use: that is either to explore possible consequences of an action before taking that action or as embedded parts of a system to aid in routine decision making [Pidd 1999]. As mentioned earlier, the pioneers of SSM use models to help them reflect on some of the human interactions in systems use [Checkland 1981].

Overall, our exploration for a model to represent software engineers' motivation aims at deriving some form of external representation of the dynamics of how software engineers are/can be motivated. Such a model will be important to the people who manage software engineers in terms of helping them to manager software engineers better.

2.2 How do you construct a good model?

Where the approach to model building is that of making a partial representation of the real world, Pidd suggests that model builders should adhere to some basic principles as outlined below:

- Models are used as tools for reflective thinking, so they should be built simple so that they can facilitate careful thought. Models that are complicated lend themselves to poor interpretation. Since the object of the model is to facilitate thinking, model builders are encouraged to "build simple and think complicated"

[Pidd 1996, 1999]. However, simple models should not mean small models. Simplicity is derived from many factors one of which is transparency. The transparency of a model can be achieved through the careful design of the problem being studied. In this respect a properly defined problem brings transparency to the resultant model of its solution. In latter sections, we would explore how our research strategy defined the problem of software engineers' motivators and how the resultant solution to that problem resulted in our model of motivation.

- Models should be developed gradually, starting with simple assumptions. If possible, adopt a modular approach to modeling so that should certain components of the model be found to be erroneous, they can be replaced without having to replace the entire model.

- The problem to be modeled must be decomposed into simpler components. This principle is similar to the last one discussed above, however the difference is reflected in how the problem itself has been defined. Powell [1996] suggests that this approach is basic to Western science. Powell, however, intimates that the difficulty is in knowing the most useful components to create. In later sections, we show how this principle has informed the formation of our research strategy.

- The use of data in model development should be for the purpose of parameterising and testing the data. Model developers should not rely on data to drive the model: *"the model should drive the data, not vice versa"* [Pidd 1996].

- Make use of analogies, metaphors and similarities at the early stages of model development to facilitate understanding of the simple models that are suggested. This can be done through the use of the modelers' own experience or drawing from the experience of previous tried and tested work. Pidd suggests that this process helps the modeler to identify and therefore couch their model in a proven logical structure. In the latter sections, we demonstrate our use of this principle by comparing the model developed in this work with some of the early work done in the same area of modeling software engineers' motivators.

- Dispense with the notion that modeling is a rational and linear process. Pidd describes the actual process of building a model as "muddling through". In fact, Willemain [1994] recounts that expert model builders

> *"Develop their models not in one burst, but over an extended period of time . . . guided by analogies, drawings and doodling . . and in each case starting small and adding"*

2.3 Concepts of Motivation

In this section we provide an overview of the concepts that we expect a good model of motivation to draw upon. These are the concepts that we expect our model of motivation to reflect. Classic motivation theories are explained in terms of either process theories or content theories where a process theory of motivation is one where the phenomenon of motivation is realized through a series of stages, whereas a content theory explains motivation as a phenomenon that is realized in one stage.

There are eight theories that specifically addressed how people are motivated within an organizational context. The following are overviews to these theories.

2.3.1 Job Characteristics Theory

The Job Characteristics Theory [JCT] states that there are certain key characteristics present within a job that makes it motivational to practitioners [Hackman and Oldham, 1976]. These key characteristics are classified into five 'core dimensions', which are Skill variety, Task identity, Task significance, Autonomy and Feedback. The extent to which these five job dimensions motivate practitioners is dependent on his need for personal growth and development; Growth Need Strengths (GNS) [Hackman and Oldham, 1976].
The basic tenet of the JCT is that a practitioner will experience internal motivation and satisfaction if his GNS is matched by the Motivational Potential Score (MPS) of the job he does. Optimum internal motivation and satisfaction is achieved when a practitioner's GNS is matched with the appropriate MPS in a job.

2.3.2 Stimulus Response Theory

Stimulus Response Theory (SRT) describes the activities that modify behaviour [Skinner, 1976]. These activities are termed stimuli. According to SRT there are two types of stimuli: punitive and rewarding stimuli. The theory explains that punitive stimuli are easier to apply and do have the effect of producing the required responses in the short term. However, rewarding stimuli, which are more difficult to apply and require more ingenuity to devise, tend to have a longer-term effect in inducing the correct responses from subjects [Skinner, 1976].

2.3.3 Equity theory

Adam's Equity Theory [1963], in an organization context, is concerned with how to make employees feel equitably treated in an organisation. It states that the inputs that people bring into an activity or organisation, that is their experience, education, skills and seniority, should be matched by the outputs that is what they get from that activity or organisation, which are salary, recognition, opportunity for achievement etc. According to the equity theory, practitioners are not necessarily satisfied by the balance between their own set of output and inputs, but will continue to compare that balance with that of other practitioners within their department, company or industry [Adams 1963].

2.3.4 Need theory – Maslow

Maslow's hierarchy of needs roughly translates that different types of needs motivate people at different stages in their lives [Maslow, 1954]. Such needs manifest in a hierarchy where physical needs are at the bottom of the hierarchy and self-actualisation comes at the top. Maslow's theory suggests that a person pursues these needs in a sequence so that, for example, a person's social acceptance needs will not dominate him until most of his security needs are met [Mata Toledo and Unger, 1985]. This way, as a person's needs are satisfied, new ones emerge to motivate his behavior.

2.3.5 Need theory – McClelland

McClelland's needs theory identifies three motivational needs: achievement, authority and affiliation [McClelland 1961]. It states that each individual has a combination of these needs in various levels of strength. So that the individual whose need mix is strongly biased by affiliation will tend to be a more objective in order to increase their opportunity for bonding with the most number of people. This could be manifest either in the work place or at home. Similarly, an individual with the strong achievement need tends to be drawn to situations that allow him to constantly challenge himself by setting goals and needing the feedback to allow him to assess his achievement of these goals. According to McClelland, such individual will generally find security and financial rewards as less motivating than say responsibilities and feedback.

The interesting distinction between McClelland's theory and Maslow's is the absence of the hierarchical structure on which these needs manifest. However, it is possible to assume that with time, an individual's needs mix will change and assume different profile depending on where he finds himself.

2.3.6 Motivation-Hygiene Theory

Herzberg's motivation hygiene theory classifies factors that motivate practitioners into two distinct sets: Extrinsic factors and intrinsic factors.

- Extrinsic factors are those that are external to the job that practitioners do. For example peer relationships, company policy and pay. Herzberg suggests that these factors are necessary in order to stop a practitioner from feeling dissatisfied with his work, but do not on their own motivate a practitioner internally. They just maintain a practitioner in his job.
- Intrinsic factors, on the other hand, are the primary determinants of motivation and satisfaction. These are the factors that are directly intrinsic to the work a practitioner does, for example the job itself, responsibility, recognition and achievement. These factors motivate a practitioner, internally, in his job.

2.3.7 Goal setting theory

This theory states that goals that are hard to achieve, when accepted, lead to better performance by the people doing them than goals that are easy to achieve [Locke 1968]. However, in order to do this, the goal needs to be

very well defined, made specific and measurable, and feedback provided so that the person tackling the goal will know when it is achieved

In a nutshell, an individual is motivated by challenging work when he knows exactly what is expected of him and can access feedback as to how well he has done.

2.3.8 Expectancy Theory

Much of Vroom's [1964] expectancy theory is based around the notion that a individual's motivation to engage in certain activity is predicated by the degree or amount of positive outcomes that he expects from this activity. Hence the term "Expectancy.

Even though all of the eight theories describe different characteristics of motivators, the underlying concepts are not necessarily mutually exclusive, because the two basic categories of motivation: content and process theories are not mutually exclusive. So that what is classified as a process theory, can be composed of two or more content theories. For example, it is widely recognised that the JCT - a process theory - is made up of two content theories of Herzberg and Maslow. So that the motivation potential described in the JCT model are the intrinsic factors in Herzberg's motivation-hygiene theory.

In the next section, we discuss how our research methodology embodied the above principles of model building in the construction of the model presented here.

3. Research method

We analysed the research problem in relation to two main question areas:

- How are software engineers motivated?
- What models of motivation exist in software engineering to explain how software engineers are motivated?

These two question areas provided us with five research questions, which we introduce later. We conducted a SLR to ascertain the state of the literature in relation to the two areas identified. In the process, we expanded our knowledge of the area of motivation and models of motivation in software engineering. We used the results of the SLR to establish the gaps in the thinking and work in the area of modeling a software engineer's motivators. Then we constructed a model of a software engineer's motivation from the results of the question that explores how a software engineers is motivated. We further refined this initial model with the findings from our analysis of the other models of motivation in software engineering. In so doing, we followed broadly the principles of good model building as prescribed by Pidd [1996, 1999] and Powell [1995] and Hodges [1991], amongst many, and described earlier. It is imperative to emphasize here that the chronological order of the model construction process is not strictly as described above, however, overall, our construction process reflects the approach that we have described. We then evaluated this model in terms of its theoretical efficacy by showing how it reflects classic concepts of motivation.

In the rest of this section we present an overview of the process of the systematic literature review and the strategy formulated to extract the model.

3.1 Overview of Systematic Literature Review

We conducted a Systematic Literature Review of software engineers' motivators, using the guidelines suggested by Kitchenham [2004]. The following are those guidelines [Kitchenham 2004]:

- Identify the need for a systematic literature review (MoMSE(cfs) 2005)
- Formulate review research question(s)
- Carry out a comprehensive, exhaustive search for primary studies
- Assess and record the quality of included studies
- Classify data needed to answer the research question(s)

- Extract data from each included study
- Summarise and synthesise study results (meta-analysis)
- Interpret results to determine their applicability
- Write-up study as a report

We formulated the following set of research questions:

RQ1: What are the characteristics of Software Engineers?
RQ2: What (de)motivates Software Engineers to be more (less) productive?
RQ3: What are the external signs or outcomes of (de)motivated software Engineers?
RQ4: What aspects of Software Engineering (de)motivate Software Engineers?
RQ5: What models of motivation exist in Software engineering

The rest of the procedure followed in the SLR can be found in [Beecham et al 2006].

3.2 Overview of model construction process

The model construction process adheres to the general principles of model development discussed earlier, so that even though the actual process did not specifically follow the stages prescribed by Pidd, in retrospect, the approach that we took adheres to the principles that Pidd prescribes. We discuss our process in the context of each of Pidd's principles, described above. In this process, we were chiefly concerned with coming up with a model that was derived out of a clearly defined problem. This problem is "what motivates software engineers?"

Pidd et al suggest that a good model must be simple. Such simplicity can be derived from how transparent the model turns out to be. Transparency, in turn, can be achieved by how well defined the problem is. In our model construction the problem we explore is tightly defined by the research questions formulated for the SLR. RQ1 to RQ4 provide us with a clear definition of the building blocks to the overall question on what motivates software engineers. The aim is that the solutions to the questions would provide the rationale for and the parameters for building the model. The overall structure of the model itself will be derived from similarities of other tried and tested work.

The rationale for building our model, in light of the research that indicates that models are important in helping us think and reflect about a phenomenon before acting, is to establish whether such a model already exists. In the context of this research, this line of enquiry is catered for by the inclusion of Research Question 5, above, in the SLR. The answers to RQ5 should help us understand what the gaps are in the area of models of software engineers' motivation.

The next stage is to simplify the problem by decomposing it into simple components. This process involves a lot of knowledge of the problem area. This requires that in formulating questions that will provide solution to the line of enquiry, we also needed to know how that problem area could be decomposed into simple modules. In doing this, we are also ensuring that should certain components of the resultant enquiry be found to be erroneous, they can be replaced without having to replace or re-do the entire model.

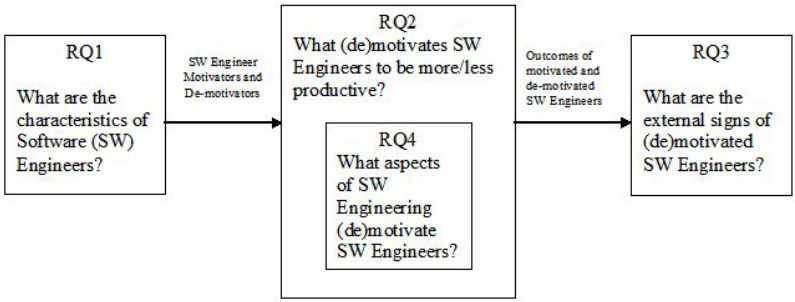


Figure 1: Framework of research strategy for SLR [Beecham et al 2007]

This simplification through decomposition is provided by the breakdown of the research enquiry into software engineers' motivation into four research question. Figure 1 illustrates this point.

The resultant data from the chain of enquiry represented in Figure 1 allowed us to parameterise the components of the individual modules of the emerging model and the overall structure of the model was then derived from analogies and similarities with other tried and tested work on generic concepts of motivation and by analysing other models of motivation in software engineering. We explain the tried and tested work and other models of motivation in Section 5, when we present and discuss the model.

The final stage in the model construction exercise was to ascertain how the resultant model reflected both classic and conventional theories of motivation. We did this by explaining the components and the dynamics of the model against what is already known and published on motivation.

The next Section presents the results of the SLR.

4 What motivates software engineers?

In the following section, we present findings from the systematic literature review to the Research Questions 1 to 5 presented earlier.

4.1 What are the characteristics of Software Engineers?

Table 1 presents results from the research question "what are the characteristics of software engineers?"

	Software Engineer Characteristics	% in studies
Ch.4	Growth oriented (e.g. challenge, learn new skills)	9%
Ch.6	Introverted (low need for social interaction)	8%
Ch.11	Autonomous (need for independence)	8%
Ch.1	Need for stability (organisational stability)	5%
Ch.3	Achievement oriented (e.g. seeks promotion)	4%
Ch.12	Need for variety	4%
Ch.14	Need for challenge	4%
Ch.16	Need to be sociable/identify with group/organisation	4%
Ch.2	Technically competent	3%
Ch.5	Need for competent supervising	3%
Ch.8	Need for feedback (needs recognition)	2%
Ch.10	Need to make a contribution (job is worthwhile)	2%
Ch.13	Marketable	2%
Ch.15	Creative	2%
Ch.7	Need for involvement in personal goal setting	1%
Ch.9	Need for Geographic stability	1%

Table 1: Software Engineer Characteristics [Beecham et al 2007]

Table 1 shows that the SLR reports a software engineer to be *growth oriented,* i.e. he is an individual who likes challenges and likes to learn new skills. It also shows that the SLR reports a software engineer to bean *introverted* individual with low need for social strengths. The SLR also reports a software engineer to be *autonomous*, *creative* and *technically competent*.

4.2 What (de)motivates software engineers to be more (less) productive?

Table 2 presents results from the research question "what motivates software engineers to be more productive?"

	Motivators in Software Engineering	% of studies reporting
M.17	Identify with the task (clear goals, personal interest, know purpose of task, how it fits in with whole, job satisfaction; identifiable piece of quality work)	21%
M.10	Employee participation/involvement/working with others	17%
M.4	Career Path (opportunity for advancement, promotion prospect,	16%

	career planning)	
M.6	Good management (senior management support, *team-building, good communication)	16%
M.3	Variety of Work (e.g. making good use of skills, being stretched)	15%
M.7	Sense of belonging/supportive relationships	15%
M.1	Rewards and incentives (e.g. scope for increased pay and benefits linked to performance)	14%
M.12	Recognition (for a high quality, good job done -different to M1 which is about making sure that there are rewards available).	12%
M.2	Development needs addressed (e.g. training opportunities to widen skills)	11%
M.11	Feedback	11%
M.15	Technically challenging work	11%
M.16	Job security/stable environment	11%
M.18	Autonomy	9%
M.8	Work/life balance (flexibility in work times, caring manager/employer, work location)	7%
M.21	Making a contribution/task significance (degree to which the job has a substantial impact on the lives or work of other people)	7%
M.5	Empowerment/responsibility	5%
M.19	Appropriate working conditions/environment/good equipment/tools	5%
M.14	Trust/respect	4%
M.13	Equity	3%
M.9	Working in company that is successful (e.g. financially stable)	2%
M.22	Sufficient resources	2%

Table 2: What motivates software engineers [Beecham et al 2007]

Table 2 shows that some of the most widely reported motivators of software engineers from the SLR are the ability to *identify with the tasks*, *employee participation*, *career paths*, *good management*, *variety of work* and a *sense of belonging*.

4.3 What are the external signs or outcomes of (de)motivated software engineers?

Table 3 presents results from the research question "what are the external signs of motivated software engineers?"

External signs of motivated and de-motivated software engineers	% of studies
Ext1: Retention	11%
Ext2: Project delivery time	2%
Ext3: Productivity	5%
Ext4: Budgets	1%
Ext5: Absenteeism	1%
Ext6: Project Success	1%

Table 3: External signs of (de)motivated software engineers [Beecham et al 2007]

The most widely reported outcome of motivated software engineers is *retention*. Our SLR reports that staff retention is improved in companies where software engineers are motivated. Other external outcomes are improvements in *project delivery time* and *productivity*, adherence to *budgets*, low *absenteeism* and improved *project success*.

4.4 What aspects of software engineering (de)motivate software engineers?

Table 4 presents results from the research question "what aspects of software engineering motivates software engineers?"

Motivating Aspects of software engineering field	% of studies
Asp1: Problem Solving (the process of understanding and solving a problem in programming terms)	3%
Asp2: Team Working	2%
Asp3: Change	4%
Asp4: Challenge (Software Engineering is a challenging profession and that in itself is motivating)	4%
Asp5: Benefit (creating something that is of benefit to someone or enhances well-being)	3%

Asp6:	Science (making observations, identifying, describing, engineering, investigating and theorising, explaining a phenomena)	2%
Asp7:	Experiment (trying something new, experimentation in order to gain experience):	2%
Asp8:	Development practices (Object Orientated, XP and prototyping practices)	2%
Asp9:	Software process/lifecycle – Software development, project initiation and feasibility studies, and maintenance (note maintenance was also found a de-motivating activity)	1%

Table 4: Motivational aspects of software engineering [Beecham et al 2007]

By this research question, we aimed at identifying the inherent characteristics in software engineering that made the discipline motivating to its practitioners. Table 4 shows that the variety of aspects reported from SLR ranged from *problem solving*, *change*, *challenging* nature, *science* to the *experimental* aspect of the discipline.

4.5 What models of motivation exist in software engineering?

Table 5 provides a summary of the different models of motivation and job satisfaction developed specifically for the Software Engineering industry as identified in our SLR. The classifications of model types show that some models fall into more than one of the categories listed below.

Models of motivation	Frequency (# of studies)
1: Job Characteristics Model (JCM) of Software Engineer (SE) Motivation (development, enhancement or validation)	10
2: Models focusing on Software Engineer Job Satisfaction	6
3: Models of Open Source Developer SE Motivation	3
4: Models of leadership influence on SE motivation	3
5: Model drawing on expectancy theory, goal-setting theory, and organizational behaviour specific to the software development process	1
6: Model of Task Design influence on SE motivation	1
7: Model of Career Progression influence on SE motivation	1
8: Social support influence on Software Engineer turnover	1

Table 5: Models for motivating SEs [Beecham et al 2007]

The following is a brief discussion on some of the key issues of these models. A more detailed exploration of the models is presented in [Sharp et al 2007].

Almost all the models draw on and build upon some classic theory of motivation, however, none of the models substantially encapsulates all the factors that underpin motivation. In effect, none of the models reflects all the concepts of motivation expressed in the literature as presented in Section 2.3.

Models presented by the Open Source studies appear to be more pertinent due to their focus on what motivates practitioners to participate in the activities of software engineering, but they do not take into consideration the particular characteristics that orientate software engineers towards that discipline in the first place. Other models that look at leadership influence on motivation are simple and as a result are a good example of what models should be, as suggested by Pidd [1996]. However, they ignore many of the core job characteristics of software engineering. The models on Job Diagnostics Survey and the other offshoots of the JCT are more useful models because these models focus on motivation as a factor of the nature of the job itself. However, these models, like the Open Source models, do not take into account factors like software engineers' characteristics. Whilst Career Progression and Social Support Influence on Turnover models tend to concentrate mainly on factors external to core job characteristics.

Overall, we suggest that though these models above provide valuable insight into software engineers' motivation, they are disparate and do not encompass all the factors that our understanding of the literature leads us to expect of a model on motivation.

5. Model construction process

In this Section, we demonstrate the process of putting our model of software engineers' motivation together. We show a stepwise approach adopted from the scientific management and operational research literature and discussed under the research methodology Section. We also evaluate the model in terms of how it reflects classic theory on motivation.

5.1 Assembling the model

In Section 3, we presented the framework used to explore the issues in this research. This framework summarised the research strategy used to frame our research questions. From this framework, we were able to evolve a model of motivation, based on the answers to the Research Questions 1 to 4. We present our model of motivation as a series of stages in a process. In this model, for example, we classify a software engineer's motivation as two sets of factors: aspects inherent in the job that software engineers do, for example the problem solving nature of software engineering and some general factors, which can also be sub-categorised into extrinsic and intrinsic factors as illustrated in Figure 2.

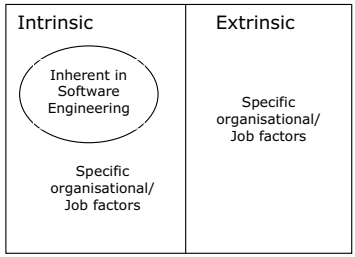


Figure 2: Factors that motivate software engineers

Our model also presents a set of characteristics of software engineers. It shows that these characteristics are mediated by individual personality traits and environmental factors.

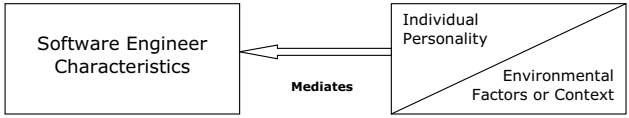


Figure 3: Factors affecting Software Engineers' Characteristics

We suggest that these characteristics orientate software engineers towards their motivators, particularly, the set of inherent job characteristics: Figure 3

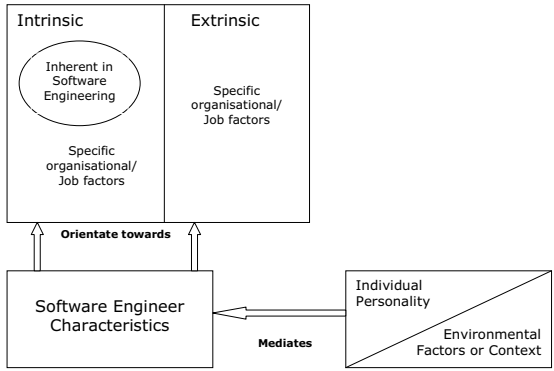


Figure 3: Software Engineers' Characteristics orientate towards motivators

Figure 4 shows that software engineers' motivation leads to external outcomes like staff retention, increased productivity and reduced absenteeism.

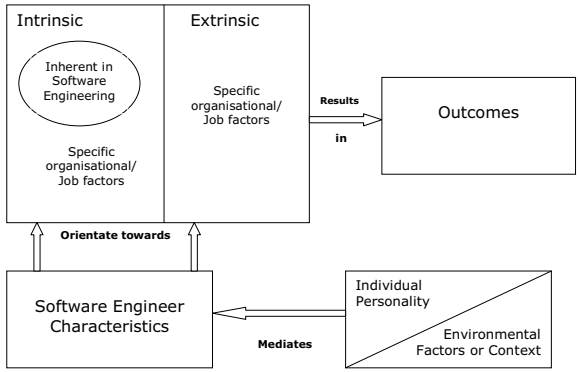


Figure 4: Motivators result in external outcomes

Having established the overall structure of the model, we are now able to parameterise the model with data from the SLR that we have presented in Section 4. In Figure 5 we show the parameters of the abstract component in Figure 2. These parameters are the data from the set of motivators found in our SLR.

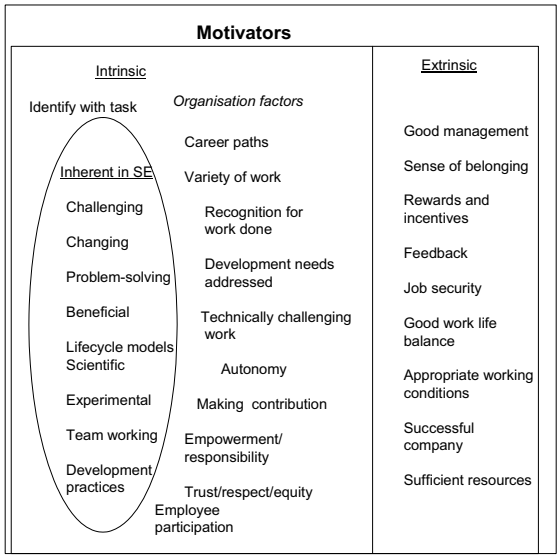


Figure 5: Parameters of motivators of software engineers

Figure 6 also shows the parameters for the Characteristics component in Figure 3.

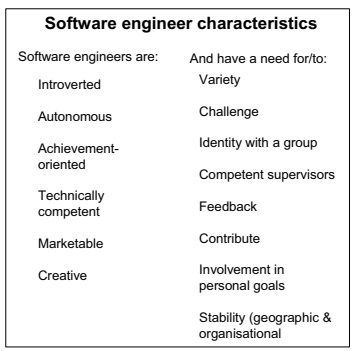


Figure 6: Parameters of Characteristics of software engineers

5.2 Refining the model

A final stage in our model construction phase was to compare our model with the other models of motivation in software engineering in order to refine the structure and dynamics of our model. Our analysis of the structure of these other models showed that the relationships between components are more complex than Figure 4 suggests. For example, we were able to discern that contextual factors have a direct effect on motivators and how effective they are. It also became clear that the balance between organisational intrinsic and extrinsic motivators and the

motivators inherent in software engineering have an effect on software engineers' characteristics, and their reactions to different motivators. The full
discussion of these finding are presented in [Sharp et al 2007]. The results of this analysis led us to refine our initial model. Figure 7 presents the refined model.

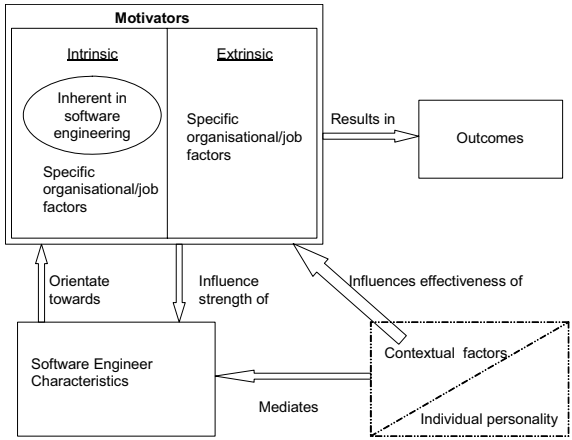


Figure 7: Refined model of motivation in software engineering

Overall, the model we present in Figure 7 is a process model because it explains the dynamics of motivation in a series of steps and stages as shown earlier. We suggest that this is so because the majority of studies in our SLR from which we have derived the above model, concentrated more on process theories than content theories. However, the notion of a process theory is not mutually exclusive from that of a content theory, so the model we have will also reflect some content theories of motivation as will be demonstrated next.

6 Theoretical efficacy

In this Section we evaluate the model that we developed in terms of theoretical efficacy. A model that attempts to represent the dynamics of how software engineers are motivated must reflect some generic concept of motivation in order to be theoretically sound. So in this section, we evaluate our model in the context of how it reflects classic motivation theory. This evaluation stage is the final step in our model construction process, as discussed earlier.

The following sections reflect the level to which the classic motivation theories are reflected in the model we have presented.

6.1 Theories predominantly reflected in our model

In this section we discuss the three theories that are predominantly reflected in our model:

- Hackman and Oldham's JCT
- Herzberg's Motivation Hygiene theory
- Task Design concepts of Expectancy and Goal setting

Hackman and Oldham's JCT

One of the most interesting observations from this work is how closely our model of motivation relates to Hackman and Oldman's JCT. Our model suggests that software engineers have certain characteristics that are mediated by their individual personalities and the environmental context, the latter being nature and type of job

and the cultural settings. These characteristics, depending on which bit of the literature you pay attention to, make software engineers out to be a distinct group of people. As a distinct group, software engineers are motivated by a set of factors, presented in Figure 5. The result of this motivation is increased productivity, job retention, lower absenteeism, and better quality work.

We suggest that the JCT, which has been widely used in models on job characteristics, described earlier, is very aptly reflected in the model that we have developed and presented here. This is not surprising because most of the studies in the SLR have drawn on the JCT and similar variations of that theory. However, our model appears to have extended the original JCT by introducing a characteristics quotient. Software engineer characteristics are absent from Hackman and Oldman's model, or indeed, Couger and Zawacki's revised version. Our model further adds depth to Couger and Zawacki's revised version and identifies particular factors that contribute towards the respective facets in that version.

In Figure 8, we map the intrinsic motivators identified in our SLR and presented in our model with the core job characteristics of the JCT.

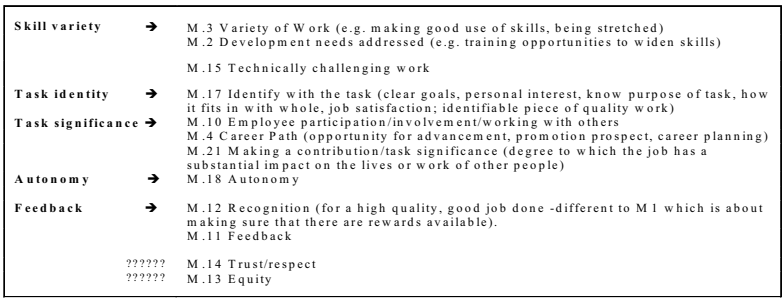


Figure 8: Mapping intrinsic motivators from SLR to core job dimensions of JCT

Figure 8 shows that with the exception of *equity* and *trust/respect*, all the intrinsic motivators from our findings map perfectly onto the job characteristics as specified in the JCT. This shows that our model draws from other tried and tested concepts and also that our attempt enhances that particular concept by introducing two new characteristics of *equity* and *trust/respect*. We discuss the effect of these new characteristics in later sections.

Herzberg's Motivation Hygiene theory

Another theory that is prominently reflected in our model is the Motivation Hygiene theory. Table 6 presents and categorises the motivators identified from our SLR into Intrinsic versus Extrinsic motivators as defined by Herzberg in [Couger and Zawacki 1980].

Intrinsic factors		Extrinsic (hygiene) factors	
M.17	Identify with the task (clear goals, personal interest, know purpose of task, how it fits in with whole, job satisfaction; identifiable piece of quality work)	M.6	Good management (senior management support, *team-building, good communication)
M.10	Employee participation/involvement/working with others	M.7	Sense of belonging/supportive relationships
M.4	Career Path (opportunity for advancement, promotion prospect, career planning)	M.1	Rewards and incentives (e.g. scope for increased pay and benefits linked to performance)
M.3	Variety of Work (e.g. making good use of skills, being stretched)	M.11	Feedback
M.12	Recognition (for a high quality, good job done -different to M1 which is about making sure that there are rewards available).	M.16	Job security/stable environment

M.2	Development needs addressed (e.g. training opportunities to widen skills)	M.8	Work/life balance (flexibility in work times, caring manager/employer, work location)
M.15	Technically challenging work	M.19	Appropriate working conditions/environment/good equipment/tools
M.18	Autonomy	M.9	Working in company that is successful (e.g. financially stable)
M.21	Making a contribution/task significance (degree to which the job has a substantial impact on the lives or work of other people)	M.22	Sufficient resources
M.5	Empowerment/responsibility		
M.14	Trust/respect		
M.13	Equity		

Table 6: Intrinsic versus extrinsic factors that motivate software engineers

In repeated studies like those of Fitz-Enz [1978] and Couger [1988] that have tested Herzberg's theory on software engineers, it has been reported that software practitioners have generally rated intrinsic factors higher than extrinsic factors, even though in these studies there have generally been fewer intrinsic than extrinsic factors cited. The results of our model shows that slightly more intrinsic than extrinsic factors have been cited as motivators for software engineers. We suggest that our SLR and its resultant model express Herzberg's theory in terms of the presence of two different factors: intrinsic and extrinsic in software engineers' motivation. We also suggest that our model supports findings of subsequent software engineering studies using that particular theory.

Task Design Theories

Task Design theories form the building blocks of process theories. For example, Hackman and Oldham's JCT is made up of the concept of expectancy – where an individual's willing ness to engage in an activity is predicated by the amount of positive outcome he expects to get from it – and goal setting, where an individual is motivated to do challenging work, so long as such work is clearly defined and constant feedback is provided of his progress. This is evidenced by the way some of the key facets of the JCT have been formulated, with task significance and identity and feedback taking prominent roles as core job characteristics.

In the model that we have extracted from the SLR, we are able to expand on this notion of task design being the building blocks of process models by providing more evidence of how expectancy and goal setting come together to formulate the model presented in Figure 7.

- Expectancy

Expectancy is where an individual's willingness to engage in an activity is predicated by the amount of positive outcome he expects to get from it. In order to ascertain what could be said to be a positive outcome for software engineers, we must first look at how they are characterised. In our model software engineers are characterized as autonomous, needing variety, needing challenge and needing feedback amongst others. In that same model the set of factors that motivate software engineers are given as the inherent challenging nature of software engineering and the general motivators of autonomy, variety of work and feedback . In the context of the theory of expectancy, we can explain these two facets of the model as:

i. Software engineers are motivated to do software engineering because they expect to get a positive outcome from it due to its challenging nature. A feature that appeases their need for a challenge
ii. Software engineers will be motivated to do work if it guarantees them autonomy, variety of work and feedback. These general outcomes will generate a positive outcome for software engineers because these outcomes will assuage those particular characteristics of software engineers.

Our model shows that being aware of what the characteristics of software engineers are, we are able to devise tasks/jobs that would make them pursue that task with the expectation of gaining positive outcomes. Hackman and Oldham's JCT is similarly designed, however our model gives more insight into the type of factors to implement, because it also takes into account the characteristics of the people for whom this model is developed. And when these factors are designed into a job/task, they make the jobs more appealing to software engineers due to the personal positive outcome that software engineers expect.

- Goal setting

The concept of goal-setting is where one is more inclined to do challenging work, so long as such work is clearly defined and constant feedback is provided of progress. This concept is reflected in our model because our model suggests that:

- software engineers are motivated by software engineering because it is changing and challenging, amongst other inherent characteristics
- However, these motivator sit side by side with work with which they can identify and from which they can receive feedback
- Implying that software engineers are motivated towards the discipline because it is challenging and changing, but this motivation must not sit in isolation since the job must be clear to them and they must receive feedback of their performance from doing that job.

Our model arranges these sets of motivators into a coherent set to reflect the concept of goal setting in job/task design.

6.2 Moderately reflected theories

Two sets of theories are moderately reflected in our model:

- Adams' Equity theory
- The Needs theory of Maslow and McClelland

Equity Theory

In an organisational context, Equity theory depicts that the inputs that individuals bring into their work, must match the outputs that they receive from that work. Whereby these outputs are not only in the form of remuneration, but also authority, responsibility and opportunities. From this perspective, but without sufficient background to ascertain overall demographic profile of the software engineers in the SLR, we are able to recognise that a majority of the factors cited as motivators are also, in fact, responses to practitioners' characteristics. For example, software engineers who are characterized as "Needing competent supervising" cite "good management and senior management support" as a motivator.

We can also explain the concept of "equity" through its literal meaning and suggest that software practitioners are motivated by equitable treatment in their jobs. One particular study cited "latitude" as a motivator, where practitioners responded that it is not just the application of motivation factors that matter, but also the equal application of these factors. This particular translation does not necessarily reflect Adam's Equity theory, per se, but can be said to extend the JCT with some hitherto unaccounted for elements like "equity or latitude". Gambil et al [2000] did something similar by including the concept of "equity" into a comprehensive model of task design. We suggest that our model accounts for the concept of equity in the same way.

Needs Theory: Maslow and McClelland

One of the fundamental observations from our findings is the temporal effect on motivators. Rubin and Hernandez [1988] suggested that software engineers who are relatively new to their jobs tend to be more motivated by hygiene factors. However, with time, the profile changes and the longer software engineers remain in their work, the less important the hygiene factors become to them.

The above observation is a typical expression of Herzberg's motivation-hygiene theory. But it is also an indication of how Maslow's needs theory manifests. As software engineers mature in their jobs, the immediate factors like the physical and job security are replaced by the more intrinsic factors like self-esteem that the nature of the job itself provides.

Similarly, this temporal effect can be interpreted by way of McClelland's needs mix theory, whereby software engineers who are relatively new to their jobs would posses a need mix that is strongly biased towards affiliation because they may want to optimise their opportunity to bond with more people. Whereas engineers who may have been in their positions longer will have needs mixes that are more biased towards achievement and or authority.

Our model fails to show this transition or mix in needs, but identifies a collection of motivators that, together, account for all the various types of factors that will influence practitioners at every stage in their development.

6.3 Theories least reflected in our model

The least reflected theory in our model is the Stimulus Response Theory. Our model indicates that out of all the motivators identified in our SLR, possibly only one motivator could be said to be a punitive motivator. We make reference to *Rewards and incentives* (e.g. scope for increased pay and benefits linked to performance)". Having established this, we can still argue that this factor could be considered a rewarding motivator because in classic motivation terms punitive motivators are factors like *threat of redundancy* - which will make practitioners work harder to ensure their place in the job - or *tight deadlines*, *performance related pay* and *orchestrated competition*.

So how does our model reflect Stimulus Response Theory in relation to software engineers? We cannot be certain, but can only state that our model confirms that software engineers are predominantly motivated by rewarding motivators [Baddoo 2001.

7 Conclusion

In this paper we have presented a model of software engineers' motivators. Our previous work suggested that no rational model of software engineers' motivation existed. The few that exist make a fairly disjointed use of classic motivation theories. In this work, we have conducted a comprehensive review of studies on software engineers' motivators and extracted a model of motivation from this review.

Our model shows that software engineers are motivated by two sets of factors, intrinsic and extrinsic motivators, where a subset of intrinsic motivators are aspects inherent in the job that software engineers do, for example, the problem solving nature of software engineering. Our model also shows that factors that orientate software engineers toward these particular motivators are their characteristics and that these characteristics are mediated by individual personality traits and environmental factors. Our model shows that the environmental or contextual factors can have a direct effect on the effectiveness of motivators. Also, that the balance between organisational intrinsic and extrinsic motivators and the motivators inherent in software engineering have an effect on software engineers' characteristics, and their reactions to different motivators. Finally, our model shows that the external outcomes of software engineers' motivation are benefits like staff retention, increased productivity, and reduced absenteeism.

The dynamics of the model presented here and the constituents of the components of the model strongly reflect three classic motivation theories of Job Characteristics, Motivation Hygiene and Task Design. Our model also moderately reflects Adams' Equity theory, because the model parameters include the factor of equity as a motivator. We argue that even though our model fails to directly show a transition in needs or mix in needs, it identifies a set of motivators that can be said to reflect the Needs theories of Maslow and McClelland. The least reflected theory in our model is the Stimulus Response theory. Overall, we suggest that this strong reflection of many of the classic motivation theories in our model gives the model the theoretical efficacy that has been wanting in many models of software engineers' motivation.

We are, however, aware of some limitations to this model. Firstly, it imperative to state that because most of the findings from the SLR are based on studies that used the Job Characteristic Theory, it should not be surprising that the JCT is strongly reflected in our model. Secondly, we do observe that a majority of the studies from the SLR have looked at motivation from an organisational management perspective and in the process have not adequately captured some of the technical and personal aspects of motivation. So this, too, could be a limitation of this model. Finally, we do appreciate that the model presented here is partly a result of answers to research questions that were taken from individual studies. Collating data through this method disjoints the theme and provides less of an insight into the coherency of the responses in the individual studies.

Overall, we suggest that our model serves as a valuable starting point for managers wanting to understand how to get the best out of software engineers, and an individual wanting to understand his own motivation, or who is embarking on career choice. We suggest that it also provides a platform from which subsequent researchers can base their empirical studies, thereby providing a well-founded basis on further motivation work.

Acknowledgements

This work was supported by the UK's Engineering & Physical Sciences Research Council under grant number EP/D057272/1.

References:

Adams, J. S., 1963. Towards an Understanding of Inequity, *Journal of Abnormal and Social Psychology*, 67, 422-436.

Baddoo N, 2001. Motivators And De-Motivators In Software Process Improvement: An Empirical Study, in *Faculty Of Engineering And Information Sciences* p 259, University Of Hertfordshire, Hatfield.

Beecham, S., Baddoo, N., Hall, T., Robinson, H. and Sharp H., 2006. Protocol of a Systematic Literature Review of Motivation in Software Engineering, *Technical Report No. 452* School of Computer Science, Faculty of Engineering and Information Sciences, University of Hertfordshire.

Beecham, S., N. Baddoo, T. Hall, H. Robinson, and H. Sharp 2007/8, Motivation in Software Engineering: A Systematic Literature Review. *Journal of Information and Software Technology*, Elsevier (accepted manuscript).

Brooks Jr. FP, 1995. *The Mythical Man Month: Essays In Software Engineering*. Addison-Wesley, Reading, MA.

Checkland, P. B., 1981. *Systems Thinking, Systems Practic*e, John Wiley and Sons, Ltd, Chichester, England.

Checkland, P. B., 1995. "Model validation in soft systems practice," *Systems Researc*h, Vol. 12, No. 1, pp. 47–54.

Couger J. D and Zawacki R. A., 1980. *Motivating And Managing Computer Personnel*. John Wiley and Sons.

Couger J. D, 1988. Motivators And Demotivators In The IS Environment. *Journal Of Systems Management* 39 (6):36-41.

DeMarco T and Lister T., 1987. *Peopleware - Productive Projects And Teams*. Dorset House.

De-Souza D. D., 1998. Managers' Motivation for Using Information Technology. Industrial Management & Data Systems 98 (7): 338-342

Fitz-Enz J., 1978. Who Is The DP Professional? *Datamation* September:125-128.

Gambill, S. E., Clark, W. J., et al., 2000. Toward a holistic model of task design for IS professionals, *Information and Management* 37 (5): 217-228. Elsevier.

Hackman, J. R. and Oldham, G. R., 1976. *Motivation through the design of work: test of a theory*. New York, Academic Press.

Hall, T., S. Beecham, N. Baddoo, H. Robinson, and Sharp H., 2007, Motivation Theory in Software Engineering *ACM Transactions on Software Engineering and Methodology (TOSEM),* (accepted manuscript).

Hodges, J. S., 1991. "Six (or so) things you can do with a bad model," *Operations Researc*h, Vol. 39, No. 3, pp. 355–365.

Jenkins, G.M., and Youle, P.V., 1971. *Systems Engineering,* Watts, London.

Kitchenham, B., 2004. Procedures for Performing Systematic Reviews, Keele University and National ICT Australia Ltd.**:** 1 - 28.

Locke, E. A., 1968. Toward A Theory Of Task Motivation And Incentives, Organisation Behaviour And Human Performance, 3, pp 157-189.

Maslow A., 1954. *Motivation And Personality*. Harper & Row, New York.

Mata Toledo R. A. and Unger E A., 1985. Another Look At Motivating Data Processing Professionals. *Computer-Personnel* **10 (1)**:1-7.

McClelland, D. C., 1961. *The achieving society*. Princeton: Van Nostrand.

Nadler, D. and Lawler, E., 1983. Quality Of Work Life: Perspective And Directions, *Organisational Dynamics*, 11 (3), pp. 20-23

Pidd, M., 1996. Five Simple Principles of Modelling, *Proceedings of the 1996 Winter Simulation Conference,* Pages:721-728

Pidd, M., 1999. Just Modeling Through: A rough Guide To Modeling, *Interfaces*, 28(2), 118-132

Powell, S. G., 1995. "The teacher's forum: Six key modeling heuristics," *Interface*s, Vol. 25, No. 4, pp. 114–125.

Rubin, H. I. and Hernandez E. F., 1988. "Motivations and behaviors of software professionals" Proceedings of the ACM SIGCPR conference on Management of information systems personnel , College park, Maryland, United States 62-71. ACM Press

Sharp, H., Baddoo, N., Beecham, S., Hall, T., and Robinson, H., 2007. Models of motivation in software engineering, *Information and Software Technology*, in press

Skinner B. F., 1976. *Walden Two*. Macmillan, New York.

Standish, 1994. *The CHAOS Report*, The Standish Group

Tomayko, J. E. and Hazzan, O., 2004 *Human Aspects of Software Engineering,* Charles River Media,

Vroom, V. H.,1964., *Work And Motivation*, Wiley, New York.

An Agile Software Process Successfully Implemented within an expanding Research and Commercial Organisation

Frances Cleary Grant, Phelim Dowling and Catherine Kehoe
TSSG Waterford Institute Of Technology
Carriganore, Cork Rd, Waterford
{fcleary,pdowling,ckehoe}@tssg.org
http://www.tssg.org

Abstract
Working in research or commercial development? Want to advance your new innovations on the road towards a marketable product? This is a natural progression and a well developed Software Process management strategy suitable for providing improved quality and realistic results is a necessity in a high demanding product development environment. Incorporating the use of agile practices, tools and quality management in this process help smooth the transfer of innovations from the research side to the commercial side. Having a verification and validation process in place, leads to a greater chance of actively feeding on state of the art research work and progressing it beyond the conceptual phase towards a marketable product. This paper will highlight the strategy of the TSSG research and development organisational approach to improving their software process and how they successfully incorporate this process on a daily basis as they work towards successful, stable and highly marketable end products.

Keywords
Verification and Validation, Software process, Agile, test driven development

1 Introduction

TSSG (Telecommunications Software and System Group){1} is an expanding research and development organisation based in Waterford, Ireland. It is divided into two primary divisions the Research Division and the Commercial (i.e. Innovation and Development) Division. The TSSG's main area of research is communications software services encompassing emerging architectures for management of complex telecommunications and Internet systems as well as next generation service development and deployment. TSSG constantly strives towards improving all processes they adopt in the day to day running of their activities. One large aspect of these processes is the highly achieving and effectively run Verification and Validation (V and V) Group that implement on a daily basis an effective agile software process to further enhance the overall quality of the communication software services they are validating. This paper will provide you with an insight into the activities required to actively implement such an agile software process, such as

- The software process model used in TSSG
- Improvements in coding standards, agile development
- Improvements in product quality standards
- Tools and software costs
- Case study of success within TSSG using this agile software process.

All great marketable products and applications initially start as a thought or concept, but the process of adopting and guiding these innovations from basic research to commercialisation is a stringent process. TSSG has many years of experience in both state of the art research and commercialisa-

tion, and are productively transferring ideas and innovative concepts from their research division to their commercial division. Having a concise and clear agile software and V and V process in place helps set the required standards between these divisions and works towards easing the transfer of knowledge process. Promoting collaboration and collective agreement is core to the success of the overall process, with all stakeholders being actively involved from day one. The TSSG has defined a specific SPI ethos, to help them focus on actual objectives and aims {2}. These include the following main points.

1. Work towards promoting capability, where project teams are fully capable of fulfilling the processes and practices.
2. Always work towards promoting and encouraging a team approach to building software applications i.e. all team members are involved from day one of the project ensuring successful collaboration and collective agreement.
3. Promote the function of Software Process Improvement at all levels of the organisation by identifying the value it brings at each level.
4. Build high quality, maintainable products.
5. Implement process and practices that add value and increase productivity.
6. Quality is owned by all team members, promoting collaboration amongst team members.
7. Continuously improve practices and processes from lessons learned to increase productivity and to ensure software process improvement.
8. Promote knowledge sharing, Usability and Accessibility.

2 Intial Groundwork Steps

In an attempt to identify the best possible software process to adopt, we initially conducted a review of the major development methodologies such as waterfall, spiral, CMMi, DSDM, SCRUM and XP{3}. Based on this research it was concluded that pragmatism is the key, and it is imperative not to slavishly follow a specific method as each project has its own unique demands. A software process adopted is domain neutral. Whilst we in TSSG are applying it to the communications domain, it is also applicable to many varied software development environments. In all cases so far, we have determined that the best fit for producing quality software is to apply a dual approach of lean/agile principles combined with full system test. We recommend the use of some of the following Extreme Programming (XP){4} practices in the list below.

- Release planning determines schedule
- Active Customer Involvement
- Simple design and Prototype to eliminate risk
- Short (usually 2 week) iterations and 3 month release drumbeat
- Test Driven Development - Unit and Acceptance
- Continuous Integration
- Optimize late
- Daily stand-up meeting
- Pair Programming
- Aggressive refactoring
- Ubiquitous automation (three strikes and automate)

The TSSG EII (Evaluate, Inform and Implement) approach to software process improvement has helped guide and implement the correct practices and activities to ensure a successful, re-

alistic and improved operational agile software process.

{Evaluating} the overall goal of an individual project is a must in order to clearly identify from the start the projects needs and requirements as it will progress through the software implementation and validation process. If the project is already an existing project, then the current processes and practices being implemented must be analysed in order to fully determine their suitability or areas where improvement can occur. Otherwise if it is a new project on its maiden journey through the process, then an evaluation if the existing organisations processes and practices will be productive for this project is required.

The next step of the TSSG approach highlights the importance of communication so that all project participants are working on a common playing field, this is completed during the *{Inform}* step. Based on valuable experience obtained over many years the validation and verification (V and V) team in TSSG actively provide advice on the architecture, applications and technologies to use during the projects lifespan that would be best suitable towards obtaining the most productive end result.

The final and most important step in the TSSG approach to Agile software process involves *{Implementing}.* The first and foremost step in the phase is the training and teaching of the involved engineering teams to help them embrace and understand the value that the software process improvement brings to an engineering team and project. This involves the training of teams to follow the recommended process and practice in order to achieve the software process improvement. By encouraging these engineering teams to embrace and understand the value that software process improvement brings to supporting the management team and organisation. This helps improve the overall uptake of the processes. It is important to promote *{Capability}* and not dependability. When initially performing an assessment of a suitable software process to adopt for a certain organisation there are approximately 5 main assessment areas that need to be taken into consideration. The first phase of this assessment would take into consideration an assessment of the organisations management, identifying the organisations structure and customer base. The assessment would then progress towards assessing the organisations product feasibility, providing a clear view of the feasibility process the organisation adopts, and timelines defined. Having this initial information would then enable you to progress towards the main part of the Engineering Management assessment where the following sections would be analysed

1. Engineering Management assessment (i.e. development methodology, code base management)
2. Requirements Management assessment (i.e. how requirements are identified, documentation of requirements, traceability of requirements)
3. Configuration Management assessment (i.e. source control system and management, release procedure)
4. Build Management assessment (i.e. build process, automation, involvement of test cases in build process)
5. Verification and validation Management assessment i.e. test responsible, types of testing
6. Defect Management assessment (i.e. prioritising defects, who responsible for defect raising, tracking, lifecycle of a defect)

Other main assessment areas would also need to take into account the process and project management to ensure risk assessment and project planning assessment would be incorporated into the overall analysis of the organisation to work towards identifying a suitable software process.

3 Agile Development Lifecycle

The agile methodology {5} is the main development method used within the commercial side of the TSSG incorporating the use of agile tools and agile practices into the software engineering process

adopted. The main concept is the application of 'lean' thinking to software development. To help achieve this we apply a set of technologically neutral principles and practices for software development.

The high level milestones for all projects are tracked in a Master Project Plan. This enables the management team to effectively plan and resource projects across the whole organisation. Requirements and User Stories for an individual project are generated and managed in the project planning tool, Xplanner {6}. A project is generally planned in 3 month cycles or Engineering Releases (ER). Each ER is then subdivided into 2 week iterations (IR). Each IR is kicked off with an IR Planning meeting where all project participants meet to discuss the outcomes of the last IR, expectations and plans for the upcoming IR and any other general project issues. A demo of the functionality verified to date may also be executed. Throughout each IR, continuous builds are executed each time code is checked into the repository. This will run the associated unit and acceptance tests. A nightly build is also generated and this may run additional load tests if appropriate. The V and V team will take the nightly build and deploy it to the independent test environment. This environment is then used to perform system testing, i.e. functional, automated, usability, load, performance, etc. on an ongoing basis. Defects are logged in our customised bug tracking tool Bugzilla, and are tracked through our defect management process.

The completion of each ER is preceded by two iterations of dedicated system test and bug fixing. Once Engineering Release Feature Set Completion Date (ERFSCD) has been hit (generally 4 weeks before the planned ER end date) a week of full system test is executed followed by a week of engineering bug fixing. Verification and validation is then repeated for another week to regress the defects fixed by engineering. At that point, assuming that there are no open defects and no other major risks, red flags or open actions against the project, the ER will be approved by V and V. The last week of the cycle involves deployment where the product goes live and all marketing activities take place.

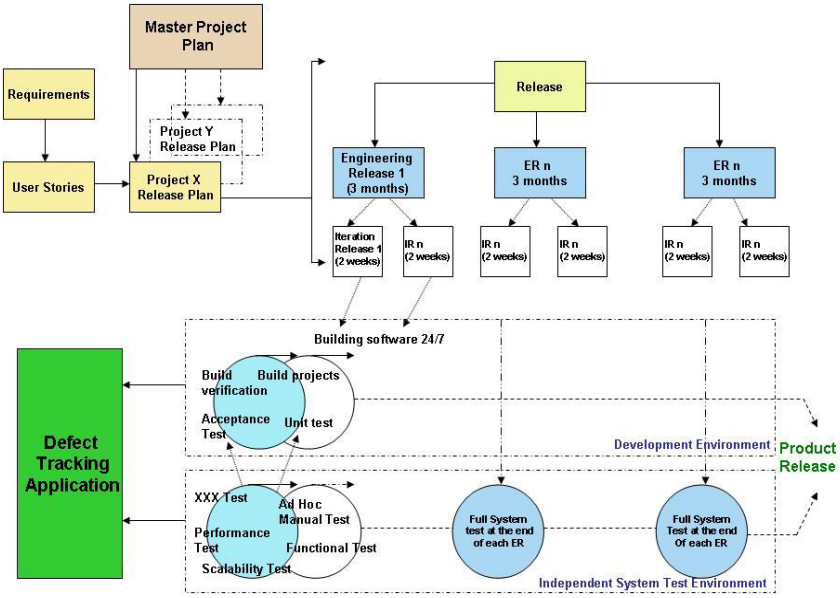


Fig 1. Agile Development TSSG

4 Verification and Validation

The verification and validation (V and V) group and their processes are intertwined with the software process improvement, with the V and V group ensuring that customer software requirements are implemented completely. The validation team take very hands on and direct approach with the customer, engineering and management. A high level of communication with the customer is very important, leading towards a greater level of trust and understanding on both sides. Having a number of procedures in place that keep the project deadlines in order and to ensure the overall quality of the product being developed leads to the opportunity for projects to adapt and modify them in order to improve the code quality of their system's development and help rethink the process of software engineering.

Within TSSG the validation Processes consists of the following main activities

1. Bi-Weekly Iteration Planning meetings
2. Code reviews
3. Usability Sessions
4. Regular Engineering Releases
5. Rigid Quality standards
6. Daily Team Stand-Up Meetings
7. Defect Management Process

{Bi-Weekly Iteration Planning meetings}: Bi-Weekly meetings involving a member of the test team, team manager and representative of customers company, provide an open forum for discussion surrounding open issues and action points from the previous weeks meeting highlighting critical activities and prioritising workloads. During such meetings a detailed mapping of a bi-weekly roadmap and work completion plan are identified and tracked, ensuring that all open and outstanding issues are raised and addressed on a bi-weekly basis to track progress. Through the involvement of the participating customer, this demonstrates to the customer on a bi-weekly basis the level of progress and also provides the customer with a certainty for queries to be resolved and potential changes that may have to be accommodated.

{Code Reviews} periodically take place typically when milestones are reached and are most commonly in the format of desktop reviews. Generally they are performed by programmers who are external to the testing and development team to ensure no code knowledge or bias exists. This is deemed an extremely good practice that ensures good programming practices are maintained and that the quality of the code is maintained.

Incorporating **{Usability Sessions}** into the process generally occur following each iteration, and this usability session occurs in house. The main purpose of the usability session is to ensure that the iteration has produced something tangible and can be operated independently. At various iterations when particular parts of the application have been developed the customer is invited to these sessions to test run the system and provide valuable input to the development team. This provides an opportunity to gain a feel for the system and to spot potential uses that the system has not catered for.

Within TSSG **{Regular Engineering Releases}** are actively incorporated into the software process, with releases of work on a 3 month schedule. Each release is subsequently broken up into 2 week iterations. Having these short iterations in place, they are used to pick apart the system in a 'divide and conquer' approach. Each iteration provides tangible outputs and can be used as a yardstick for progress. Having these short iterations can help build confidence within the team and within their customer, knowing that the system is coming together stage by stage. These regular engineering releases also help in the planning for unexpected additions if the need arises.

{Rigid Engineering Quality standards} must be applied during this validation process, to ensure that the engineering team are building quality into the product as opposed to V and V team testing quality into the product. The team responsible for the testing implementation have very high standards that must be met. The following are an example of some of the standards adhered to

1. A build script which takes in all the tests developed is run nightly.

2. The benchmark for a successful test run on a build is 100 percent test cases run successfully, with 90 percent line coverage and 100 percent branch coverage running successfully.

3. Anything below this figure is unacceptable.

Quality checks also occur when a team member tries to update the project with some new code. The new code along with the entire project is subjected to an intense testing procedure and if the 90 percentage barrier is not reached, the code is not accepted for a build.

The use of **{Daily Team Stand-Up Meetings}** has been very beneficial. Every morning the team has a 10 minute stand-up meeting. This meeting is an informal team meeting where the previous day's issues can be raised so as to avoid waiting until the next weekly meeting. Issues that are blocking are noted so that solutions can be reached and development can proceed. It consists of a short summary of the previous day's goals and an outline of the coming days work is expressed to all.

A stringent **{Defect Management Process}** is followed to safeguard the ongoing quality of the product being developed. Regular reports are run against Bugzilla to allow analysis of trends in the quantity of bugs being raised, the severity of these bugs and the status of the bugs as they progress through the process. Regular defect review meetings are held to discuss the open defects, plan fixes, etc. At the beginning of each iteration the resolution of open defects is given higher priority above all other work.

Based on experience within the TSSG of implementing these processes in the validation and verification phase, a clear and visible improvement in the overall quality and standards of coding has increased leading to more trustworthy and reliable end products of a very high level.

5 Tools Used in Agile Software Process

The utilisation of the correct and most useful tools to benefit the software process need to be taken into account, it is extremely important that the proper tools are used to support the projects going through the process. Where possible in TSSG we try to utilise open source software to support our testing needs, with the benefits of this being lower costs, continuous releases and a thriving support community. Agile software process adoption requires a toolset for the process and specific software development domain. Broadly speaking TSSG employ process supporting tools and development supporting tools. For process supporting tools, these support the software development process and help with scheduling/planning and estimation and also reflect back information from developers to managers. Development supporting tools are more technical in nature but are still aligned to process adoption and also reflect back metrics to managers (although some interpretation is required). Whilst these tools will vary dependent on implementation platform and language, the function that each tool provides remains the same.

TSSG Process Supporting Tools	
Process	**Tool**
Release Planning	MS Excel/MS Project
Iteration Planning	Xplanner
Defect Tracking	Bugzilla
Knowledge repository	Project wiki
Continuous integration	CruiseControl

Table 1 Process supporting Tools

Table 2 conveys a list of development supporting tools used within TSSG Organisation. Trying to keep software costs low is a high priority, and by utilising suitable open source applications, the overall software costs near zero, apart from MS Excel/MS Project licence cost. We successfully

build all of the required infrastructure using high quality, free, open source tools. Many free open source tools exist for closed platforms (i.e. .NET)

TSSG Development Supporting Tools	
Family	Tools utilised
Source Control	CVS, Subversion
Build Tools	Ant, Maven,NANT
Unit Testing	Junit, httpUnit,Cunit
Code Analysis	Checkstyle, Jalopy
Documentation	Javadoc
Test Frameworks	FIT, Extractor, Abott
Code Coverage	Cobertura
Architecture	Structure 101

Table 2 Development Supporting Tools

6 Test Driven Development

Test driven development {7} {8} has been effectively used within project lifecycles within the testing domain for some time now. Through the use of iterative sessions incorporating new functionality through the implementation of test cases prior to/in parallel with development work, this leads to an efficient process and is a major positive contribution towards software process improvement. The following is a list of some of the main guidelines that we abide with for the test driven development phase.

- Developers write unit tests in parallel with application code
- Developers write at least one Acceptance test (end to end test) for each user story
- Automated test coverage measurements to help enforce test quality - typically 90 percent line coverage
- Educate developers on how to write effective tests
- Pair programming helps keep tests effective
- Additional code reviews focus on testing
- Any reported defect MUST result in a new unit or acceptance test against the code-base before the issue can be closed (positive feedback loop)

As well as following the test driven development methodology, by also incorporating the use of continuous integration leads to more realistic outputs and helps towards increasing the level of overall quality. These continuous integration activities involve some of the following activities within TSSG

- Define an automated build script that
    - Executes unit tests
    - Executes acceptance tests
    - Measures build stats (test coverage..)

- Use continuous integration tool to execute the build every time code is checked in.
- Alerts via e-mail and webpage if build fails.
- Developers make small incremental commits to the codebase.
- Build must pass, otherwise development halts until it is fixed.
- Zero tolerance policy on severity 1 and 2 defects - development halts until fixed.

7 Case Study Of TSSG Agile Software Process in ISERVE/INIFINITIM Project

The ISERVE/INFINITIM{9} project was kicked off in September '06 and stands as a fine example of how the verification and validation process within the TSSG can lead to the successful development of a highly marketable product. The team, which was led by a Product Centre Manager within the TSSG comprised up to 8 developers, a V and V engineer and a Customer representative. The goal of the team was to produce an innovative Session Initiated protocol (SIP) based Instant messaging (IM) client for PC and mobile (InfinitIM) and a ground-breaking IP Multimedia system (IMS) service platform (iServe).

The first ER was planned to culminate with a fully functional system for presentation at the 3GSM conference in Barcelona in February '07. 10 iterations were mapped out including a schedule of when each piece of functionality would be delivered for system test. Continuous and nightly builds were generated from early in the first iteration with email alerts to the team on code coverage misses and test failures. System test started in iteration 2 when some of the early features were delivered. Daily stand-ups and biweekly iteration planning meetings kept all team members synchronized on short and medium term plans and expectations. A number of usability sessions were organised where feedback from the customer and other invited participants was taken on board. A code review was executed by the TSSG Chief Architect and his recommendations were implemented. The original Engineering Release Feature Set Completion Date (ERFSCD) of 5th January 2007 was missed by 3 days - not bad for the aggressive schedule outlined at the start, and as with all good plans, enough leeway was incorporated into the schedule to absorb such a delay and reschedule a new engineering release. A week of full system test was followed by a week of dedicated bug fixing. This 2 week cycle was then repeated at which point the product was released for ER1,followed by a week of validation and verification and a week of deployment where the product goes live.

When the dust settled after 3GSM a full review of the ER was completed with feedback from all participants. This led to the implementation of certain improvements in our process. Although some of the faces changed, the project continued to evolve in 2007 through ER2 and ER3 with the same standards of quality maintained through process adherence. ER4 is soon to be kicked off while a version of iServe is about to enter trial with a major Telco.

Similar verifications and validation processes were adopted for other projects within TSSG, such as Muzu {10}, FeedHenry {11} and Zinadoo {12}. Having this agile software process and validation process in place has proven its importance in such a working environment, limiting the number of bugs that manage to pass undetected through a projects lifecycle, in the end working effectively towards a more stable and reliable product.

8 Conclusion

As the agile software process used within TSSG promotes Capability as opposed to dependability, this in turn leads to effective project teams that are capable of fulfilling the defined processes and practices. By adopting a team approach to building the software applications, ensures that all team members are involved from day one of the project.

Promoting the function of software process improvement at all levels of the organisation is a must in order to identify and highlight the value it brings to each level, all the while increasing quality aware-

ness. By raising the need for increased level of quality, helps focus their vision and implement the activities required to build high quality maintainable products the whole time working towards implementing process and practices that add value and increase productivity.

It is imperative to encourage like minded thinking that quality is owned by all team members, ensuring this frame of mind leads to each team member adopting responsibility for the overall quality of the product. We have learned that we must continuously improve practices and processes from lessons learned to increase productivity and to ensure software process improvement. This in turn helps promote knowledge sharing, usability and accessibility. The TSSG has seen great improvement in end products that pass through their software process and the value of this agile software to the organisation is unquestionably positive at all levels.

Positive effects of the agile software process visible by our *{Senior management}* through

- Quick Frequent delivery of features to market.
- Significantly lower volume of defects reported by Customers.
- Visibility into the development process.
- Metrics available on various aspects of SPI.

{Middle management} also see various benefits of adopting our agile software process such as :

- Cost of fixing issues reported by customer in field is less than if we were following a waterfall approach.
- Easy to introduce new features.
- Visibility into the development process.
- Metrics available on various aspects of SPI.

{Development, test and support} also experience benefits of the agile software process such as:

- Maintainable,repeatable, reliable, quality code base.
- More defects identified through process and practice followed by development team.
- Team is empowered to succeed.

Based on the research {13} and experiences of implementing this software process in the TSSG organisation, the defect quantity level showed a clear decrease in the number of defects highlighted during the project validation phase, this in turn proves the valuable contribution that the agile software process has to an organisation towards improving the overall quality of a product.

Literature

{1} TSSG {http://www.tssg.org}

{2} Beck, et al., *Manifesto for Agile Software Development,* http://www.agilemanifesto.org/

{3} XP {http://www.extremeprogramming.org/}

{4} Beck K, Extreme programming explained ISBN: 978-0201616415, Published 1999

{5} Scott W. Ambler,Agile Database Techniques (effective strategies for the agile software developer), ISBN: 0-471-20283-5,Published: 2003

{6} Xplanner {http://xplanner.org/}

{7} TDD {http://www.agiledata.org/essays/tdd.html}

{8} Beck K, Test driven Development By example, ISBN-13: 978-0321146533, Published 2002

{9} ISERVE/INFINITIM {http://www.aceno.com/projects.html}

{10} Muzu { http://www.muzu.tv/}

{11} FeedHenry { http://www.feedhenry.com/ }

{12} Zinadoo { http://www.zinadoo.com/ }

{13} Panlabs II {http://www.panlab.net/ }

{14} ESQAT {http://www.esqat-china.net/en/ }

9 Author CVs

Frances Cleary Grant

Frances Cleary is currently working in the role of senior researcher/test engineer at the Telecommunications Software & Systems Group (TSSG) research centre. Frances has worked in European Funded projects such as Daidalos (Designing Advanced network Interfaces for the Delivery and Administration of Location independent, Optimised personal Services) and ESQAT (Embedded Software Quality Assurance & Testing). She has been involved in various test bed design & Planning, test bed deployment and test execution phases. She has also worked in various Integration Planning and conformance test execution phases throughout the lifespan of many projects. Frances is also actively involved in internal TSSG test bed initiatives and IPv6 centre of excellence related activities within TSSG. From 2000-2004 she worked for Ericsson Systems Expertise based in Athlone, Ireland during her time there she worked in the Fault Management, Access and Core telecommunication areas, as a test engineer.

Phelim Dowling

Phelim has 12 years experience in the IT industry working in a variety of roles and companies. As a technical/project lead at the Intel R&D facility in Oregon he was responsible for coordinating and executing the development, testing and deployment of numerous products into Intel's manufacturing sites worldwide. On moving back to Waterford he spent time as a Test Engineer with Waterford Technologies. Following this he acted as a consultant project engineer/manager specialising in IT project delivery in the highly regulated pharmaceutical industry. In 2006 Phelim joined the TSSG Verification
and Validation team and has worked on projects such as iServe, Muzu and FeedHenry.

Catherine Kehoe

Catherine has over 10 years experience across a broad base of the IT and telecommunications industries. Catherine joined the 3CS in 2006 as Verification and Validation Manager. Prior to this Catherine was the Verification and Validation Manager for the TSSG's commercial division at the Waterford Institute of Technology (www.tssg.org) (www.wit.ie). Catherine joined the TSSG in 2005 where she successfully implemented a Verification and Validation function across all projects in the commercial division of the TSSG.

Process Certification in Safety-critical Domains

Lic. Tech. Risto Nevalainen
Tampere University of Technology, Pori
Pori, Finland

M.Sc. (Eng) Mika Johansson
Tampere University of Technology, Pori
Pori, Finland

Lic. Tech. Hannu Harju
VTT Technical Research Centre of Finland
Espoo, Finland

Abstract

Software Certification is an essential part of qualification and licensing of digital I&C systems in high classes of safety-critical systems. Nuclear power plants have strict policy in safety, and therefore they need high-level, commercially active certification services in this area.

As a part of Finnish nuclear research program SAFIR2010, a project called CERFAS started in 2007 to define necessary software certification service for nuclear industry needs. Main areas of the service are process assessment (leading to certification) and product evaluation. Several additional modules and methods may be needed and will be developed during the project.

Project did during 2007 a state of the art report and evaluated certification schemes also in other domains. This article includes some results from the state of the art phase, mainly in process certification.

Keywords

Safety-Critical Software, Certification, Qualification, Quality, Process Assessment, ISO/IEC15504

1 Introduction

Nuclear power is an essential part of energy production in Finland. Currently there are four Nuclear Power Plants (NPP) in Finland, and one more is under construction. Additional new NPP´s are currently in feasibility phase. Of course, also intensive political discussion is going on about the future alternatives of energy production in Finland.

Current NPP´s need to be renewed so that they will be based on modern technology. Their I&C systems will be partially digital and software-based, even in safety-critical areas. It is too strong to say, that new NPP`s will be „digital", but software is essential part of their safety in future. Hardware and electronics is well understood and their acceptance mechanisms are well established. Type testing and certifications are normal part of licensing mechanism. That is not yet true for software. Therefore, a policy and selected methods are needed to evaluate, qualify and license software in current and future NPP´s.

"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". Software certification is first time an explicit topic in SAFIR2010.

Certification of software products by independent evaluation has been practiced in the software industry since early 1990s, especially in Europe and in United States. Type acceptance for equipments is required in highest safety class of I&C equipments and systems in NPP, and recommended in lowest safety classes. In the research project CERFAS, the objective is to develop facilities for flexible, supported, commercially exploitable, high quality Software Certification Service, SCS, able to certificate safety critical software for the demands in Finnish nuclear area.

All equipment in Safety Class 2 and essential accident instrumentation in Safety Class 3 shall possess a type acceptance certificate (Guide YVL 5.5). In response to this need, CERFAS-project develops facilities for software certification services primarily to demands on NPP control fields in Finland. Also for the sake of commercialization needs it is necessary to allocate the services to other national and international industry. The main purpose of the project is to develop facilities for a high level Software Certification Service. Conditions for high level services are the application of diverse expertise and effective dedicated evaluation tools. This leads into networking both in the project and in the software certification services. Certifiers must have competing products and unique features that are desired by the clients and what the other certifiers have not in long future.

The strategic objective of CERFAS is to develop facilities for flexible, supported, commercially exploitable, high quality Software Certification Service able to certificate safety critical and safety related software. The main other features to support the Service are the following:

- advanced methods for evaluation of software process and artefacts, that is, documents, code, test plans, etc.
- competence development to provide facilities for Software Certification Service.

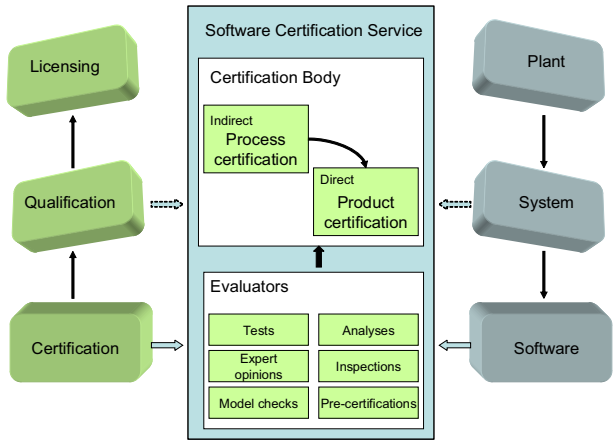


Figure 1. Elements of certification service, as defined in CERFAS project /Harju 2007/

The need for software safety certification service has been explicit for a while. State of the art phase during 2007 collected best practices both from product evaluation and process assessment point of view. First version of process certification was developed as a combination of generic quality and capability requirements vs. nuclear specific requirements in each lifecycle phase. The development consisted mainly from the following tasks:

- Evaluation of existing and evolving methods and tools for process evaluation.
- Definion of processes required by safety class 2 systems during development, maintenance and use.
- Definition of generic and nuclear specific requirements for each required process.
- Definition of evidences needed to evaluate process performance.
- Integration of the evaluation method, evidence handling and other method components as one approach for process certification.

2 *Overview of the Certification Concepts*

Certification can be defined as “the process of assessing whether an asset conforms to predetermined certification criteria appropriate for that class of asset” /IEEE 2006/. This idea of conformance with criteria is the most fundamental principle in certification.

Table 1. Main generic references and standards supporting safety class 2 certification

IEC/EN 61508	Functional safety of electrical/electronic/ programmable electronic safety-related systems. 1998.
ISO12207	Software engineering lifecycle process standard, republished in 2008
ISO15288	Systems engineering lifecycle process standard, republished in 2008
ISO15504 Part 2	Normative requirements for process assessment
ISO15504 Parts 5 and 6	Exemplar Process Assessment Models for software and systems engineering processes.

ISO15504 Part 7	Draft standard (TR2) to assess Organisational Process Maturity

In general, certification is seen in CERFAS as a service to support system/software qualification and further licensing. Various areas of methods are needed to support certification, see figure 1. Some examples are process assessment, product evaluation and use of different models and analyses. Several other current and previous SAFIR2010 projects have also contributed especially for qualification. One parallel project called MODSAFE is developing methods to validate system and software requirements by using model checking methods.

For certification, an accredited Certification Body (CB) is needed, as shown in Figure 1. Some CB´s have already a strong position in certification of safety-critical systems, for example TÛV in Germany: Type testing or other kind of Independent Verification and Validation (IV&V is typically a fundamental part of the certification. Any certification body needs a variety of services to run certification service and integrate different approaches as a coherent system.

Nuclear power industry has a classification for their safety related I&C systems. Safety-critical digital I&C systems belong to safety class 2. Safety-related systems belong to either safety class 3 or 4. All digital I&C systems in these safety classes must be qualified and further licensed before commissioning. Certification is required only for safety class 2 systems.

Table 2. Main normative references for safety-critical software in safety class 2 in NPP´s

Reference	Name
Guide YVL 5.5	Instrumentation systems and components at nuclear facilities. STUK 2002.
IEC 60880	Nuclear Power Plants – I&C systems important to safety – Software aspects for computer based performing category A functions
IEC 61513	Nuclear Power Plants – I&C for systems important to safety – General requirements for systems

Software in safety class 2 I&C systems can be external (typically a COTS component), a standard element of system (typically a platform) or a unique application. Need for certification is mainly for standard software, when it is used for example as a platform for applications.

Certification is not any isolated activity in software development and licensing. Most articles see it anyway as an independent and external activity. Maybe the most advanced practices in software certification are in space and avionics industries in USA. One specific type of certificate in that context is certified software code produced by qualified or certified tools. That kind of certification is not anyway the most common type, what we found in the literature study. Most models and industrial articles refer to some generic certification service type, for example ISO9001 for organization's quality management. One other well established area is personnel certificates, for example certified software engineer or software tester examinations, given by universities or commercial institutions.

Table 3: One classification of certification services

Main types of certification services	Examples
Quality management system certification	ISO9001 certificate for quality management. ISO27002 certificate for information security.
Process certificates, according to SPICE, CMMI, IEC61513, ISO20000 etc.	Certified process profile, CMMI based process maturity level
Product certificates, against some technical standards (for example, ISO25000).	Product inspection. May include results from type tests and/or IV&V.

Measurements and analyses, typically related with process and/or product certificates. 1	Operational history analysis. Safety analysis. Software complexity. Test coverage analysis.
Personnel certificates, typically according to some Body of Knowledge (BoK) requirements.	Certified software engineer, certified tester, Lead Assessor.
Accreditation certificates. Anyway, some facilities are needed to get accreditation for software certification services, for example assessor/evaluator competence.	Outside CERFAS scope. Certification bodies need accreditation to increase creditability of their service.

Table 3 gives one way to classify certification approaches. Most relevant for this study are process and product certificates. Their scope varies a lot depending on generic or domain specific requirements. Maybe ISO9001 is a typical example of the most generic approach, and certified code according to DO-178B requirements another extreme.

Each certification type has its own certification elements. Framework in figure 1 assumes that certificate is based on some reference model, norm or set of criteria. Certificate itself is then a conformance statement against those requirements. Typically such statement is done using some methods. Some typical methods are external audits, independent verifications and validations (IV&V), reviews and inspections, code analysis and type tests. System centric certificates may include some hardware related methods, like aging tests and electromagnetic tests.

3 Common Features of Process Evaluation Approaches

3.1 Quality as a Starting Point

Chapter 3 presents generic process assessment and certification approaches, and their adaptations in some industries which have heavy safety requirements for software. The two main generic models are ISO/IEC 15504 (sometimes called also SPICE) and CMMI. Both models have been developed mainly during late 1980´s or during 1990´s, and have been also republished as "next generation" versions.

The origin of the SPICE and CMMI models is in the need to have more detailed criteria for supplier selection and process certification than is in ISO9001 model. ISO9001 reference model leads to abstract approach, and leaves lot of space for subjective interpretation. Implementations of the quality management system vary then a lot. Some domains – for example telecommunication and automotive industries – have their own ISO9001 interpretation and guidance models. Those tailoring do not include any software or systems engineering specific guidance.

ISO/IEC 15504 Part 2 standard defines two different models, which are needed to perform ISO compliant process assessments. First, we need a process reference model (PRM), which includes a set of processes and their necessary details. Typical examples of such models are ISO12207 for software engineering and ISO15288 for systems engineering. CMMI is also a generic process reference model for software and systems engineering. Secondly, we need a process assessment model (PAM), which includes a measurement scale for capability levels and a rating scheme. Additionally, ISO/IEC 15504 Part 2 has conformity and compliance criteria for any published models, so that they can be used for example as elements of certification service.

[1] We had difficulties to find generic, normative and easy metric for software quality (an even software safety) in this literature study. It may reflect the overall immaturity of software industry.

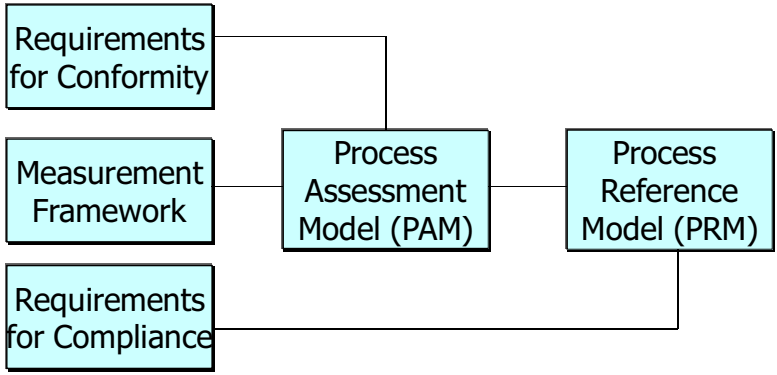


Figure 2: Concepts of PAM and PRM in ISO/IEC 15504 Part 2

Figure 4 shows how software and systems engineering specific PRM models and ISO9001 are partially overlapping and have synergic elements. They are often used also as a combined set of criteria for certification. The certificate type can be ISO9001, but covers also software and systems engineering specific requirements.

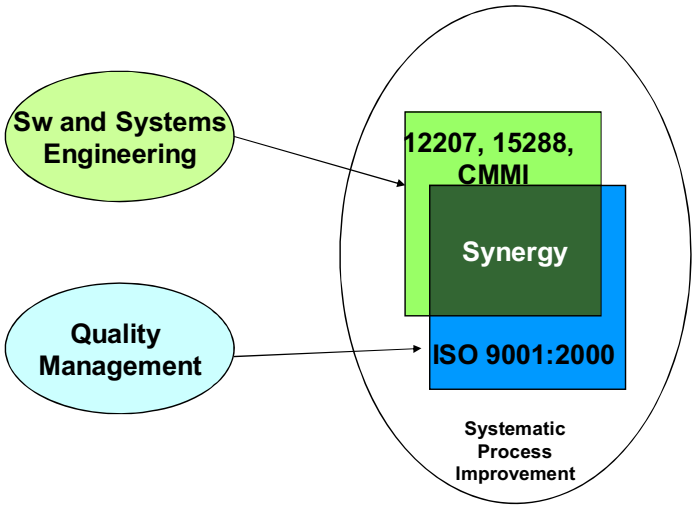


Figure 4: Potential synergy with quality management and software and systems engineering approaches

3.2 Process Capability and Maturity

Generic process reference models have been so far mostly sets of requirements, for example ISO9001, IEC61508 and IEC60880. Certification against those models is similar like conformance evaluation. ISO9001 is well known in all communities, and is a basic requirement in all safety-critical domains. IEC60880 is an example of software requirements in nuclear power I&C systems. Certificate against IEC60880 is essentially the same as satisfaction of all its requirements. Corresponding certification method is a detailed checklist. Standards are like “shall” type collections of requirements.

Process assessment models bring certification from conformance-driven requirement sets and statements to more structural approach. We can even say that this kind of new models is a new generation, compared with “shall” based references.

The main idea of existing process assessment models (mainly SPICE and CMMI) is to distinguish between process specific requirements and generic process management related requirements. Then we can isolate for example technical requirements in one set (typically called as level 1 of any process). Then we can create another set of process management and quality principles in each such technical requirement (typically levels 2 – 5 in existing models).

ISO/IEC 15504 Part 2 defines a measurement system of process capability, based on 6-point ordinal scale. The fundamental concept in the scale is growing capability of any process, to be better managed and controlled at higher levels. At higher levels the selected process is also measured and is fully compliant with business goals and needs. Process is then also more predictable, because it is better measured and has less variation in performance. CMMI model includes also a capability scale for its PRM processes, and it is quite similar as in the ISO model.

The capability level concept allows constructing a capability profile from a set of selected processes. Profile can be either based on context specific needs and goals (target capability profile) or real results or assessment (achieved capability profile). In some safety-critical domains each software safety class has its own target capability profile. Maybe the best example of target capability profile is European Space Agency´s SPEC and S4S model.

First CMM, then CMMI and very recently also ISO/IEC 15504 Part 7 define a process maturity scale. Maturity scale is also ordinal similarly as capability scale. In ISO model the concept is called Organisational Maturity. The idea of process maturity is to combine a set of processes to define a maturity level. Typical relationship is that each process at some maturity level must reach also the corresponding capability level. For example, to achieve CMMI maturity level 3, all 18 processes must also reach their capability level 3.

There are numerous other concepts of process and organisational maturity than CMMI and ISO/IEC 15504 based models. Anyway, the ISO and CMMI models are most relevant and most often used in safety-critical domains. Maybe the best known example of process maturity based qualifications and certificates is DoD policy that each software vendor shall achieve maturity level 3 to be qualified as a supplier for DoD. Nuclear power domain needs clearly a similar type of approach, and the best starting point is ISO/IEC 15504 Part 7.

The ISO/IEC 15504 Part 7 idea of organizational maturity is a generalisation of CMMI ideas [ISO/IEC 15504-7]. The basic process sets are typically software development processes, for example software specification or software design. Each domain has to define what the most relevant processes are and include them in basic set. Those processes are then improved to their higher capability level. When a defined basic set and extended set have all their processes at capability level x, also organization is then at maturity level x. Some additional rules are used to achieve maturity levels 4 and 5.

ISO/IEC 15504 Part 7 is still so new that no domain has yet defined their own basic and extended sets. It can be expected, that this standard is anyway the most popular starting point for maturity models.

3.3 Process Assessment Process

The process assessment process both in ISO based and CMMI models is fairly linear and simple, consisting of 4 – 7 phases depending on the model details. ISO/IEC 15504 Part 2 defines requirements for ISO compliant process assessment, and has more detailed guidance in Part 3. In CMMI family, the process assessment model and process is called Scampi Appraisal Model. At high level, we can see

four different subprocesses (or phases) of the assessment process both in SPICE and Scampi, see figure 6. Of course, they are defined at more detailed level in both model families2.

The initiation and planning phase defines the assessment scope, including assessment workflow, purpose and scope definition. Assessment scope is an essential element in planning, defining the processes and their target capability levels to be covered.

Data collection is essentially the same as collection and evaluation of evidences. The minimum requirement for data collection is to collect work products and to perform adequate amount of interviews. Both SPICE and CMMI models have detailed requirements for data collection in their PAM models.

The purpose of data validation and consolidation phase is to evaluate if there is enough data and evidences for rating. Consolidation is done normally over selected instances of the assessment. Most of data and evidence is also typically in instances.

Rating and reporting phase produces assessment results. As a minimum, it consists of process capability level ratings and assessment findings. Findings are classified as strengths or weaknesses. ISO/IEC 15504 Part 4 defines two basic approaches of process rating and reporting based on assessment purpose:

Improvement driven rating, looking for potential improvements and potential to achieve higher capability levels. Some space is left to subjective and expert-based opinions and interpretations of ratings and related findings, based on the improvement needs of the organization.

Capability determination based rating, which identifies a potential gap between rated and target capability levels. If the target capability is not achieved, then Part 4 has a detailed schema for gap classification and the type of risk in each process. This way to express assessment result is seen to be very useful for certification purposes. Practically, no space is left to subjective opinions and interpretations in rating and reporting in this type of process assessment.

Improvement driven and capability determination driven process assessment approaches can be combined. Combined approach is even necessary in such certification cases, when better commitment and motivation to invest in process assessment is needed.

3.4 Pre-qualification and Qualification Assessments

Process assessment can be done in several phases of software development and purchasing. Qualification is used in nuclear power industry to support the acquisition and licensing of safety-critical I&C system. TVO SWEP method [TVO2007] defines two basic types of process assessments related to qualification:

- Pre-qualification assessment, based mostly on generic software development process of the supplier and previous applications and references. The assessment instance is mostly similar type of delivery to some other customer. If the type of instance and its safety requirements are similar as in the real acquisition, the process assessment can be done for supplier selection. If the result of process assessment is near enough to target capability and the amount of gaps and their related risks are manageable, then the final contract of acquisition can be negotiated. Often also some kind of conformance with selected standards is done together with process assessment, for example to verify compliance with IEC 60880 or IEC 61513.
- Qualification assessment, based on pre-qualification findings and gaps and additional requirements of the real instance. The mode of qualification assessment is then “filling the gaps”. If any weaknesses and non-conformances still exist they need special action to be

[2] `Scampi 1.2 manual is more than 250 pages.

closed before admitting a qualification. The assessment mode is in both cases process capability determination rather than process improvement. TVO SWEP method defines also a way to combine process assessment, compliance with nuclear power specific standards and IV&V requirements as one coherent set of evidences.

4 Process Certification in Nuclear I&C Systems

High-level and capable software development process is an essential part of software quality. Many de-jure and de-facto standards and models like ISO/IEC 15504 and CMMI are developed to assess software process. The most rigid versions of these models are used to certify software process.

ISO15504 Part 5 (known as the SPICE model) is used in as the main source of process assessment. The latest published ISO standard version ISO15504 Part 5 is used as the baseline. Part 5 has all ISO12207 processes and not all of them are relevant for certification purposes. Many nuclear specific standards include quite similar concepts of processes as ISO15504 Part 5, and they are also used as normative sources. IEC60880 can be mapped with ISO12207 processes and other elements quite easily and completely.

A practical way to certify software processes in safety class 2 is to integrate generic approaches and nuclear specific approaches together. This is done mainly by combining process assessment indicators from ISO 15504 Part 5 and nuclear specific requirements from IEC 60880. It is also possible that some additional processes are needed to cover all IEC 60880 requirements, and they can be most likely expressed similarly as generic ISO 15504 processes.

So far, CERFAS project has not yet defined any specific process set from ISO12207 or ISO15288 to be certified. We developed a method called TVO SWEP during 2005 – 2006 to be used for qualification of safety class 3 software. The process assessment module of TVO SWE was a combination of ISO15504 Part 5 and nuclear specific standards. Not all SPICE processes are as relevant as others, and also the cost-effectiveness of process assessment indicates rather a short than complete list. The criterion for process selection has been alignment and integration of ISO12207 processes and related nuclear specific standards. In current CERFAS project focus is in safety class 2, and then the nuclear specific references are those listed in table 2.

The result of SPICE assessment is a capability level for each process and a number of evidences. They can be used as “load evidence” for more detailed safety analysis. SPICE capability level is not always the best way to express the real capability of each process. Therefore also a “capability index” is calculated as a ratio of evaluated practices and their sum compared to target level of the process. FiSMA has defined an extension of ISO15504 based assessment, in which the concept of capability index is defined.

5 An Example to integrate SPICE and nuclear power standards

TVO SWEP method for Safety Class 3, was based on a set of SPICE processes. The requirements from nuclear specific standards were added as subpractices to the base practices. The standard SPICE model for selected processes consisted on appr 200 base practices, and the nuclear specific requirements extended the model with more than 800 additional requirements. A few base practices were added, concerning hardware design, independence of testing and preventing common cause failures. The „size" of the assessment model in terms of processes, practices and subpractices is shown in table 4.

During the CERFAS project TVO SWEP is extended to cover Safety Class 2, where IEC 60880 is the main standard. It's requirements address both software product and software development process. For product there are many detailed requirements concerning for example architecture, programming languages and failure detection capabilities. Process requirements include for example specific type of lifecycle for development, documentation requirements and COTS qualification. Since TVO-SWEP is process oriented model, IEC 60880 requirements were interpreted from process point of view. The detailed requirements for the product presented in the appendices were excluded, since these are covered in the type acceptance tests.

In the first phase IEC 60880 requirements were first distributed into SPICE processes as subpractices. This added some 400 new subpractices into model for the level one, as shown in table 5. From the model point of view this was not exactly correct. For example, the IEC 60880 defines many requirements for the work products, for which a correct place might be in general practices or general work products in process attribute 2.2. rather than in 1.1. Other type of difficulty of this kind of mapping was to find a suitable place for quality assurance activities. A requirement to verify the design (chapter 8.8.2 in IEC 60880) could be located as a subpractice or practice in ENG.5 Software Design process, an additional requirement in SUP.2 Verification process, or add a note for Generic practice 2.2.4 Review and adjust work products in ENG.5. For the first pilot, these kind of requirements were splitted so that activities requiring indepent performer were put in to verification, validation, problem management etc. processes, and those performed by the development team were put directly in the engineering or project management processes. For the reporting purposes, it was not relevant whether the requirements were added to level one, two or three, since all processes were anyway assessed up to capability level three, and the nuclear specific requirements didn't affect the capability index of SPICE processes. The table xxx shows an example of nuclear specific requirements combined with SPICE base practices. See also column shows the most important relationships to help the assessor.

In addition, it seemed that the easiest way to include requirements related to tools from IEC 60880 chapter 14, was to create a new process. The process included couple practices to select, qualify and maintain the tool set for developing safety critical software. The security issues were other area which was not clearly presented in earlier version of TVO SWEP, but for the first pilot these requirements were embedded into old processes instead of creating a new process. The „size" of the final model is shown in table 4.

The assessment model was piloted to assess it's suitability. The target system was a radiation metering device, which had embedded software. The assessment was not purely a process assessment, but also some product specific requirements were checked at the same time to save time and effort. The model was suitable for the work, but some improvement ideas especially mainly regarding the practical issues:

- Especially the engineering processes were quite large, having at worst almost hundred subpractices in addition to SPICE base practices. The managing of a model of this size demands alot from the assessment tool from usability point.
- The division of IEC 60880 requirements and ISO/IEC 15504-5 practices were quite close to each other, but since 60880 requirements were splitted in different order, it was some times hard to find the correct requirements...

The „third edition" of the TVO-SWEP model will not have more standards or requirements in it, but the existing model will be rearranged to better match the idea of capability model of ISO/IEC 15504.

Table 4: Size of the assessment models

Process set	Processes	Base Practices	Subpractices (nuclear specific requirements)
ISO/IEC 15504-5 processes selected for TVO-SWEP	22	617	0
TVO-SWEP for safety class 3	22	621	807
TVO-SWEP for safety class 2	23	625	1277

Table 5: Partial mapping of nuclear specific requirements into base practice ENG.9.BP1.

Base practice	Subpractice	Source	Safety Class	See also
ENG.9.BP1 Develop system integration and regression test strategies.		15504-5	2&3	
	System integration plan shall be prepared and documented in the design phases and verified against the class 1 system requirements.	60880 9.1.1	2	MAN.3
	System integration plan shall be prepared sufficiently early in the development process to allow any integration requirements to be included in the design of the system and its hardware and software.	60880 9.1.2	2	MAN.3
	System integration plan shall specify the standards and procedures to be followed in the system integration.	60880 9.1.3	2	SUP.2, SUP.3
	System integration plan shall document those provisions of the system quality assurance plan that are applicable to the system integration.	60880 9.1.4	2	SUP.1
	The system integration plan shall take into account the constraints, within any set of hardware and/or software modules, made by the design of the system, of the hardware and of the software. The plan shall include the requirements for procedures and control methods covering: -system configuration control -system integration; -integrated system verification; -fault resolution.	60880 9.1.5	2	SUP.2, SUP.8, SUP.9

Base practice	Subpractice	Source	Safety Class	See also
	System integration plan shall describe types of test to be performed, test environment and acceptance criteria	61513 6.2.3.a	3	
	Integration test shall be based on a concept of stepwise integration	61513 6.2.3.b	3	

Table 6: Partial mapping of nuclear specific requirements into base practice SUP.2.BP3.

Base practice	Subpractice	Source	Safety Class	See also
SUP.2.BP3 Conduct verification.		15504-5	2&3	
	The design verification shall address: a) the adequacy of the software design specification for the software requirements with respect to consistency and completeness down to and including the modular level; b) the decomposition of the design into functional modules and the way they are specified with respect to: technical feasibility of design; testability for further verification; readability by the development and verification teams; modifiability to permit further modification; c) the correct implementation of safety requirements.	60880 8.2.2.1	2	ENG.5
	The result of the design verification shall be documented.	60880 8.2.2.1	2	ENG.5 GP 2.2.4
	The documentation shall include the conclusions and identify clearly issues that need actions, such as: - items which do not conform to the software requirements; - items which do not conform to the design standards; - modules, data, structures and algorithms poorly adapted to the problem.	60880 8.2.2.3	2	SUP.9
	The implementation verification shall include activities based on source code analysis and tests. The source code analysis is done using verification methods such as code inspection, possibly using automated tools.	60880 8.2.3.1.1	2	

Base practice	Subpractice	Source	Safety Class	See also
	Using the documentation, it shall be possible to trace each verification result back to the associated functional and non-functional requirement(s). For this purpose appropriate comments shall be contained in the source code documentation.	19265 2.5.3	3	
	All communications and interactions between the verification team and the design team, which may have a significant bearing on the verification results, shall be recorded in writing.	19265 2.5.3	3	
	All verification activities shall be comprehensively documented to enable auditing.	19265 2.5.3	3	

6 Process Certification in Some Other Selected Domains

6.1 Automotive SPICE

Automotive SPICE is the first full adaptation of ISO/IEC 15044 Part 2 requirements and Part 5 exemplar to safety-critical domain. Car and vehicle industries have already a long tradition in specific tailoring of generic models. Maybe the best known example is QS-9000, the interpretation and customization of ISO9000 series for automotive industry. See http://www.qs-9000.org/ for further details. It is originally developed by Ford, Chrysler and GM to manage their supplier relationships. Even though each element of ISO 9000 is an element of QS-9000, QS-9000 adds clauses to the majority of the ISO 9000 elements. For example, QS-9000 adds requirements for a business plan, tracking customer satisfaction and benchmarking.

Similarly, also systems and software engineering needs additional support in automotive industry, because software is nowadays the most critical factor in car safety and is also the biggest cost element in the design of new cars. The result from adaptation of ISO/IEC 15504 is called Automotive SPICE. Originally it was initiated by European car manufacturers, but is nowadays used extensively also in Japan. Some US car manufacturers also use Automotive SPICE. The main usage of Automotive SPICE is to certify software suppliers for car industry.

The current PRM version is 4.3 and PAM version is 2.3. They both are fully compliant with ISO/IEC 15504 Part 2 requirements and can then be used to claim ISO-based certificates. The published PRM model includes only normative parts of each included process, namely process purpose and process outcomes. PAM includes the necessary process indicators similarly as in ISO/IEC 15504 Part 5.

6.2 ISO26262 for Automotive Industry

Automotive industry has also another important initiative for software certification than Automotive SPICE. A consortium of manufacturers and suppliers is currently developing a full PRM model, based on IEC61508 standard. The process structure is partially based also on ISO/IEC12207. The result is called ISO26262 Functional Safety in Automotive Electronics. Key driver for development of ISO26262

are legal and regulatory requirements for safety. Industry itself sees software as a main risk for example for massive recalls of cars.

The standard ISO26262 covers both hardware and software requirements. The underlying process model is “modified V-model”. Both hardware and software has also their specific V-model based lifecycle. The set of supporting processes includes quite extensive list of analyses and assurance activities, for example safety analysis, tool verification, qualification of tools and proven in use argumentation. The plan is that ISO26262 will be in DIS phase during 2008 and will be published in 2009.

6.3 Software Certification in Space and Avionics Industries

One of the pioneer industries in software certification is space and avionics. NASA can be seen as the first major contributor in software certification; at least since late 1970´s. NASA has organized also a network of services around the certification of software for space shuttle and other devices. Independent V&V (IV&V) companies for space and avionics seem to be almost an industry itself.

Maybe the most advanced concept to certify safety-critical software is presented in DO-178B, developed by Federal Aviation Administration (FAA). The types of certificates are safety, operational and security.

The FAA certification process for commercial aircraft is such that systems are certified together on the aircraft platform through a series of flight tests. Flight-testing is the final stage of testing during certification of aircraft systems. Prior to the flight test phase, there will likely be a series of integration tests that take place in a laboratory environment. Certification credit for both hardware and software is accumulated over time during laboratory and flight-testing. Configurations of both hardware and software are monitored closely so any change impact to certification credit is minimized. Since there is such a strong coupling between hardware and software configuration, historically, final certification credit is given to the system and not a given hardware or software version.

Recent developments within the FAA have now opened the door also to certification acceptance of software independent of a hardware platform [Zelkow2001]. This notion of Reusable Software Components promises economic benefits to those that make use of the FAA's guidance.

In Europe, European Space Agency is the main player in space industry. It has developed a set of standards and models both for software development lifecycle and for quality assurance. The main concept is quality model called SPEC [ESA2001]. SPEC is mainly developed for software evaluation rather than process assessment.

Starting from the set of goal-properties and properties identified for the Space Quality Model Framework, SPEC contains the relationships among those quality properties that have to be measured, the evaluation methods that are (may be) required for supporting the collection of data for calculating metrics and the metrics themselves for evaluating the quality properties. Then the Space Quality Mode Framework is tailored according to the software criticality classes and the space application areas. SPEC can be seen as an advanced integration of product evaluation and process assessment approaches in space applications. It covers all 4 levels of safety-critical systems for space vehicles and their I&C systems.

The process assessment and certification in SPEC framework is called S4S, “SPICE for SPACE”. It is fully based on ISO/IEC15504 Part 2 requirements. Similarly as in Automotive SPICE, S4S has a separate PRM and PAM models. PRM is based mainly on ISO/IEC 12207, and is extended to cover safety requirements in space applications.

S4S model and SPEC define also a target capability level for each software class. ESA classifies software in safety class A…D depending on the criticality. In the most advanced class A, several processes need to achieve capability level 4 to be fully compliant with ESA requirements. It means that

each of those processes shall have a measurement based control mechanism to manage special causes of variation and deviations from process related business goals.

S4S model could be used in nuclear power industry as one advanced exemplar. Almost no other model has similar concept of target capability levels and profiles.

6.4 Software Standards and Models in Electro Medical Industry

Electro medical industry has similar type requirements for software safety as nuclear power, automotive, space and avionics sectors. Some standards and models use approximate synonym like medical devices or electrical medical equipment from this industry.

USA is leading in software certification for medical devices. Already in 1994, Underwriters Laboratories (UL) published the first edition of UL 2601-1, which ultimately replaced the traditional U.S. medical device standard UL 544.1. UL 2601-1 standard adopted the requirements of the IEC 60601-1:1988 standard that was the global benchmark for electrical medical equipment safety, and added requirements also from some other standards. Current version ANSI/AAMI ES 60601-1:2005 was published to minimize the deviations and to bring the U.S. document more in line with the requirements worldwide.

AAMI has published also a standard ANSI/AAMI SW68: Medical device software — Software life cycle processes [ANSI2000]. This medical device software standard uses the framework for software life cycle processes established by ISO/IEC 12207 to describe the set of required processes, activities, and tasks necessary to convey this knowledge of intention and provide this demonstration of implementation. In general this set is less than the comprehensive set defined in ISO/IEC 12207; however, where appropriate the requirements of ISO/IEC 12207 have been added to and modified.

ANSI/AAMI standard is an exemplar from ISO12207 based reference models. Software hazard management is a part of overall medical device risk management and cannot be adequately addressed in isolation. This standard requires the use of ANSI/AAMI/ISO 14971 for risk management. Risk management as defined in 14971 deals specifically with risk to safety. One portion of ANSI/AAMI /ISO 14971 pertains to control of identified risks associated with each hazard identified during the risk analysis. The software hazard management process in this standard is intended to provide additional requirements for risk control for software. This standard may be used when software is a stand-alone medical device, or an embedded or integral part of the final medical device.

Medical device development requires the execution of a number of system and subsystem life cycle processes. Requirements for these are provided in IEC 60601-1-4 for Programmable Electrical Medical Systems. When a medical device contains software, the software comprises one or more of the subsystems of the medical device. This standard provides requirements for the life cycle processes, activities and tasks required for software subsystems of medical devices. It covers the life cycle of medical device software from assignment of system requirements to the software subsystem until the software is no longer in use. This standard divides software engineering activity into two primary life cycle processes (development and maintenance) and five supporting life cycle processes. Each life cycle process is further divided into a set of activities; with each activity further divided into a set of tasks.

The other main contribution in electromedical software is standard IEC 62304: Common aspects of electrical medical equipment used in medical practice. It is also based on ISO12207 standard. Software is often an integral part of medical device technology. Establishing the safety and effectiveness of a medical device containing software requires knowledge of what the software is intended to do and demonstration that the use of the software fulfils those intentions without causing any unacceptable risks. This standard is a good example about adaptation of ISO12207 to one domain, medical devices and their software. Focus is in safety requirements. An interesting difference to some other domain specific models is more explicit and rigid concept of risk management, both as a process and as a method to avoid software failures.

7 Conclusions and Future Developments

So far, CERFAS project is starting its second year, and two additional years are ahead. The toolset for certification will be developed in several increments. In each phase, tools are used in real pilots to get feedback and to collect further needs for certification. When writing this article, the first pilot is going on and we cannot say much about the results. Main topics during 2008 will be further integration of selected methods. Integration with ISO25000 standard elements is under consideration, and some results may be seen soon.

The ultimate goal to integrate process and product evaluation approaches could be to define an evaluation module for software safety, using ISO25000 scheme for that. Current toolset of certification is implemented partially in FiSMA Assessment System GNOSIS and in Excel sheets. It is quite possible that the method will have several technical implementations in the future.

Other industries may have quite similar needs for certification and qualification as nuclear power plants. For example control of railway and metro networks and traffic, space and aviation applications, car and vehicles and electro medical devices have similar needs.

8 References

- Harju, H. Ohjelmiston luotettavuuden kvalitatiivinen arviointi (The qualitative assessment of software dependability). VTT Technical Research Centre of Finland. VTT Research Notes 2066. 2000 (In Finnish).
- Harju, H. Software Certification Service: Facilities for type approval of system software product. Research Report VTT-R-09700-07, 2007
- IEC 60880, Software for computers in the safety systems of nuclear power stations. IEC 2006.
- IEC/EN 61508, Functional safety of electrical/electronic/ programmable electronic safety-related systems. 1998.
- IEC 61513, Nuclear power plants – Instrumentation and control for systems important to safety – General Requirements for Systems. 2001.
- ISO15504 Part 5, Exemplar Process Assessment Model. 2006.
- ISO15504 Part 7, Assessment of Organisational Maturity. Draft TR, ISO 2008.
- Guide YVL 5.5, Instrumentation systems and components at nuclear facilities. STUK 2002.

9 Author CVs

Risto Nevalainen

Senior Advisor in FiSMA, part-time researcher in Tampere Technical University Pori Unit

Mr. Risto Nevalainen (Lic. Tech.) has long experience in software measurement and quality topics. He has been managing director of STTF Oy for last 10 years. His working experience include position as managing director of Finnish Information Technology Development Center during 1989-1995. Before that he had different research and management positions for example in Technical Research Centre (VTT), Technical University of Helsinki (HUT), Finnish Prime Minister's Office and Finnish Economic Planning Centre. Mr. Nevalainen has participated in ISO15504 (SPICE) standard development since beginning. He is Competent SPICE Assessor and ISO9000 Lead Assessor.

Mika Johansson

Executive Director in FiSMA, partner and consultant in Spinet Oy

Mr Mika Johansson has been working as a consultant in quality and project management areas for more than ten years. He has been Executive Director in Finnish Software Measurement Association FiSMA since 2006. In Spinet Oy, Mika has performed assessments and audits using SPICE, CMMI, ISO9001, ISO/IEC20000 and nuclear specific models since 2003.

Integrated Automotive SPICE and Safety Assessments

Richard Messnarz[1], Hans-Leo Ross, Stephan Habel[2], Frank König[3], Abdelhadi Koundoussi[3], Jürgen Unterrreitmayer[4] , Damjan Ekert[1]
[1]*ISCN GesmbH, Schieszstattgasse 4, A-8010 Graz, Austria*
Tel: +43 316 811198, Fax: + 43 316 811312, Email: rmess@iscn.com
[2]*Continental Automotive, Germany*
[3]*ZF Friedrichshafen AG, Friedrichshafen, Germany*
[4]*SQS AG, Cologne, Germany*

Abstract

In 2005 Automotive SPICE (based on ISO 15504) has been published (see www.automotivespice.com) and used in major automotive firms world wide. In parallel the topic "Functional Safety" became important due to changes in liability law and the development of IEC 61508 as an application and branch independent standard for functional safety. As a result, ISO WD 26262 a ISO draft for functional safety has been initiated classifying systems with ASIL (Automotive Safety Integrity Levels) levels and requiring additional processes, techniques, and methods to illustrate the competence for managing systems which have an impact on the loss of lives.

An Automotive SPICE assessment usually takes (for the processes defined in the scope of the German automotive manufacturing association) 4 days per project. Adding the scope of a safety assessment this dramatically increases the number of hours used in assessments. In a working group of major automotive suppliers and assessment tool suppliers we developed from 2005 – 2008 an integrated assessment approach. Portals in the above mentioned automotive suppliers already use this environment.

The working group which elaborated the methods and tools described in this paper are part of the SOQRATES initiative (www.soqrates.de) where more than 20 leading German firms collaborate in cross company task forces.

In this paper we want to explain the results of the analysis done, the assessment model applied, and what kind of reports the integrated assessment environment is producing. The results of all work (except of the proprietary assessment tools) will be made public to all suppliers by the end of 2008.

1. History and Motivation

In 2003 the SOQRATES [6] (www.soqrates.de) initiative was formed and Automotive SPICE [4] was introduced into 16 firms in Germany. At the end of 2003 the firms decided to continue (kick off funding was done by the Bavarian SW initiative in 2003) in task forces to elaborate best practices to achieve a capability level 3 in certain core competence areas, such as system and SW design, requirements management related processes, and testing related processes. Since 2003 (for 5 years now) the teams elaborated annual knowledge releases which were transformed into training materials and distributed to the partnership. (= Cross company learning cycle).

In 2005 the partnership formed a further task force with Continental, ZF, IMBUS, SQS, and ISCN to elaborate a mapping between the Automotive SPICE and the new functional safety standard, to extend the content of SPICE assessments, and to elaborate tools for assessment which allow extending the assessment scope.

The following knowledge releases have been developed:

Excel based mapping between Automotive SPICE and IEC 61508 / ISO WD 26262 [5] (this mapping will be made public by the end of 2008).
A process model to perform an assessment where an integrated team of Automotive SPICE assessors and safety assessors perform an assessment (this mapping will be made public by the end of 2008).
An integrated assessment tool set supporting the combined assessment. This will stay proprietary and will be offered commercially.

2. Applied Mapping Strategy - Automotive SPICE & Functional Safety

Automotive SPICE (and ISO 15504) defines base practices for a set of processes and capability levels 1-5 with associated process attributes and generic practices in a capability dimension (based on the measurement framework outlined in ISO 15504 – 2).

The ISO WD 26262 for functional safety describes ASIL (Automotive Safety Integrity Levels) levels A-D. ISO WD 26262 also defines a set of processes and a set of methods and tools to be used (not recommended, recommended, highly recommended) per process depending on the analysed ASIL level. The ASIL indicates the necessary measures to prevent harm to human being, related to a specific safety goal. Specific metrics are specified to scale the reliability of measures and the effectiveness of diagnosis of relevant failure modes according the defined ASIL.

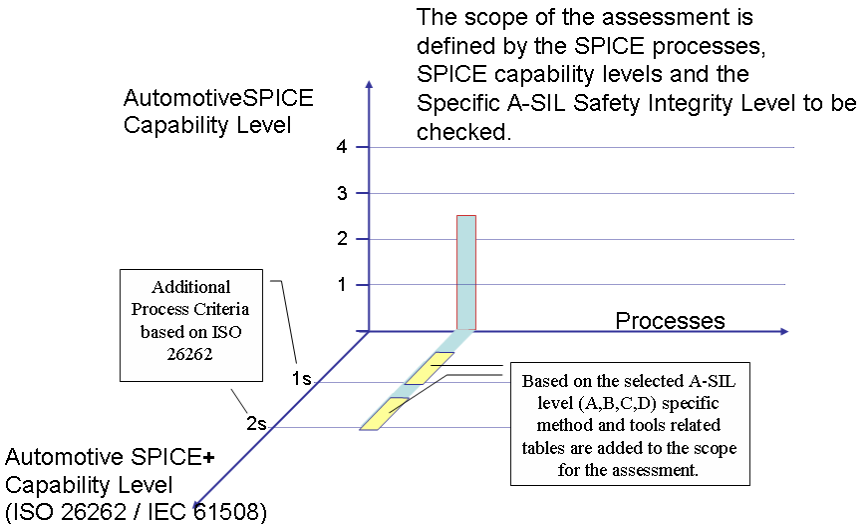


Figure 1: Integrated Assessment Framework

In the partnership the following integration strategy has been developed (see Figure 1).

To additionally cover the process and practices related requirements demanded in Functional Safety, the current Automotive SPICE has to be extended:
The text of existing base practices had to be extended.
Additional base practices (safety practices) had to be added.
Additional work products had to be added to specific processes.
Generic practices on capability level 2 of Automotive SPICE had to be extended.
Generic practices on capability level 3 of Automotive SPICE had to be extended.
To additionally cover method and tools related recommendations given by ISO WD 26262 / IEC 61508 for specific ASIL levels the method tables of the functional safety standard have been included in the assessments tools and scope.
The scope of an assessment is now defined by (ISO 15504 based) the set of processes and the selected capability levels, plus (Functional Safety based) an ASIL level to be assessed. Based on that ASIL level the proper parts of the method tables are displayed and used to interpret the practices and assess practices.
The assessment results display separate and combined views:
Assessment report just for Automotive SPICE.
Assessment report just for Functional Safety related practices and methods.
Combined assessment report integrating both.
A team assessment with a separation of ratings was necessary.
SPICE assessors document findings and do ratings and these data are displayed and stored separately.
Safety assessors document findings and do ratings and these data are displayed and stored separately.
Both data can be displayed to the team and shown in a comparative profile.
Limitation of usage
Experiences in the trials show that such an integrated assessment approach is useful to check whether specific SPICE capability levels are reached and specific safety related practices and methods are appropriately applied (based on the selected ASIL level). The integrated assessment presented in this paper focuses on processes and the use of methods, and it would not cover a product assessment (like e.g. done by TÜV) where the specific electronic and software based safety architecture and related hazard, error and risks analysis results are evaluated.

3. Mapping Results

The mapping was done based on an Excel template and required a set of 12 review workshops over a period of 2,5 years. After 1,5 years the results were reviewed by an automotive manufacturer [7] and representatives of the working group for the ISO WD 26262 which resulted in a further year to incorporate more materials into the mapping.

Figure 2 describes the general mapping approach. Figures 3,4 provide an example of how the mapping has been done in detail. The mapping itself will be made public at the end of 2008.

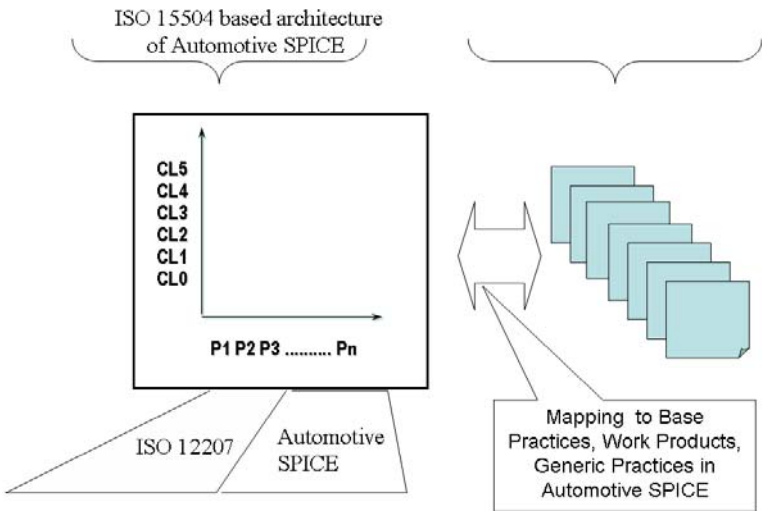


Figure 2: Mapping Approach

Overall Results of the mapping were

In approx. 60% of the cases the base practices needed to be extended. In all engineering processes additional base / safety practices had to be added.
100% of the capability level 2 generic practices had to be extended.
Five of eleven of the capability level 2 generic practices had to be extended.

The processes proposed by the functional safety standard are mapped onto Automotive SPICE processes (see filter for ENG.3 System Architectural Design in Figure 3).

Automotive SPICE	Process	Functional Safety Source	IEC 61508 Reference
ENG.3	8.8	Safety Analysis als Zusatz bei BPs	Teil 2 Kap. 7.4.7.4
ENG.3	8.9	Analysis of CCF, CMF, cascading failures	Teil 2 Kap. 7.4.3.2.2
ENG.3	3.6	Hazard Analysis	Teil 1 Kap 7.4.1
ENG.3	3.7	Functional Safety Concept	Teil 1 Kap 7.5.1
ENG.3	4.6	System Design	Teil 2
ENG.3	5.10	Overall Requirements for HW - SW Interface	Teil 2 Kap 7.5

Figure 3: Mapping Functional Safety (publications about ISO WD 26262) and A-SPICE Processes

Once we selected ENG.3 we can see that a number of Functional Safety Processes (3.6 Hazard Analysis, 3.7 Functional Safety Concept, etc.). For ENG.3 you then find a sheet in the Excel workbook where the ENG.3 System Architectural Design process has been extended in different ways (see ENG.3 System Architectural Design Work Sheet Extract in Figure 4) to cover these additional functional safety criteria.
Extended practices
Additional Work Products
New practices

A	C	D	E	F	G	H	I
Unit	Practice	ISO 15504 Outcom	ISO WD 26262 / IEC 61508 additional work products	IEC 61508 Reference	Tool / Methods table IEC	Zusatztexte in der BP	Mapping HIS
System Architectural Design	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.	1,4	functional safety requirements, non-functional safety requirements	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	Teil 2 Tab. A.16, A.17, B.2, Teil 3 Tab. A.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). NOTE 2: The objective is to meet all of the following requirements. 1) the requirements of hardware safety integrity comprising:- the architectural constraints on hardware	ENG.3.SP1, ENG.3.SP5
System Architectural Design	Identify the safety-relevant hardware and software parts. Treat all the hardware and software as safety-relevant where a safety-relevant system is intended to realise both safety and non-safety functions unless it can be shown that the implementation of the safety and non-safety functions is sufficiently independent (i.e. that the failure of any non-safety-related functions does not affect the safety-related functions).	9	safety relevant HW and SW modules			NOTE : Safety-relevant functions shall be separated from the non-safety-relevant functions, wherever feasible. NOTE : Sufficient independence of implementation is established by showing that the probability of a dependent failure between the non-safety and safety-related parts is sufficiently low in comparison with the highest safety integrity level associated with the safety functions involved. NOTE : Caution should be exercised if non-	ENG.3.SP3
System Architectural Design	Document and justify independence. Where independence is required document the method for achieving it and	9	safety relevant HW and SW modules			The control functions run independently in a parallel observation and diagnosis mode.	ENG.3.SP4
System Architectural Design	BP 2: Allocate System Requirements. Allocate all system requirements to the elements of the system architectural	1,2,8	functional safety requirements, non-functional safety requirements	Teil 2 Kap. 7.4.2.2 Anmerkung 1		This includes the allocation of a SIL level to the system requirements and system elements in case that the system	ENG.3.SP2

Figure 4: Example Extraction from the ENG.3 System Architectural Design Process

To understand the detailed mapping approach we explain how to interpret the lines (compare with Figure 4).

Interpretation of first line in Figure 4:

The original Automotive SPICE related text of Base Practice 1 for ENG.3 is “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”.
To cover the additional aspects of IEC 61508 / ISO WD 26262 it is necessary to include the following aspects:

To search for additional work products associated with the design “functional safety requirements, non-functional safety requirements”.
To consider the method tables in Functional Safety (e.g. IEC 61508 or ISO WD 26262).
To extend the text of the base practice to include the following aspects:
This includes the decision about parallel or sequential design to derive/satisfy the safety 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).
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.

Interpretation of second line in Figure 4:

The second line does not show a BP number of Automotive SPICE. It is a new safety practice which had to be added to the ENG.3 System Architectural Design related set of practices.

A new Safety / Base Practice was added with
Text : Identify the safety-relevant hardware and software parts. Treat all the hardware and software as safety-relevant where a safety-relevant system is intended to realise both safety and non-safety functions unless it can be shown that the implementation of the safety and non-safety functions is sufficiently independent (i.e. that the failure of any non-safety-related functions does not affect the safety-related functions).
Work Product: safety relevant HW and SW modules .
Notes to be considered:
NOTE: Safety-relevant functions shall be separated from the non-safety-relevant functions, wherever feasible.
NOTE: Sufficient independence of implementation is established by showing that the probability of a dependent failure between the non-safety and safety-related parts is sufficiently low in comparison with the highest safety integrity level associated with the safety functions involved.
NOTE : Caution should be exercised if non-safety functions and safety functions are implemented in the same safety-relevant system. While this is allowed in the standard, it may lead to greater complexity and increase the difficulty in carrying out system safety lifecycle activities (for example design, validation, functional safety assessment and maintenance).

In addition you can see a column HIS [7] Mapping. This refers to a non-published safety assessment method which was created as a draft by the German manufacturers association. In return for having access to this work the partnership agreed to publicise the results of the SOQRATES team to all suppliers in 2008.

In total this mapping has been done for a few hundred practices. This means that the Excel mapping is quite large and comprehensive.

4. Integrated Team Assessment Process

The assessment process is based on a team approach [1], [2], [3] (see Figure 5).

The SPICE and safety assessor work in a team. Both use a team portal system which allows both to see the others comments and ratings during the assessment. The team system is being sued already by the participating major suppliers and the SOQRATES team members.
The ratings for SPICE and the ratings for safety (extended view with additional criteria) are separately stored so that we can clearly separate between SPICE and safety interpretations.
E.g. this topic is of critical importance. Our first approach was to combine the ratings. However, soon it became obvious that …
… if we extend the scope of a base practice (e.g. see “Interpretation of first line in Figure 4”) the rating in safety scope requires more additional criteria.

… the background of the rating mechanism (SPICE assessors rather check the processes, safety assessors rather check the product structure and the methods used) is different, so that we need to keep track of a separate rating.
Reports and profiles showing coverage and capability must be generated in three different modes
Only SPICE
Only Safety
Combined
The team assessment requires sufficient interview time per process. You should calculate 2 – 3 hours per process.

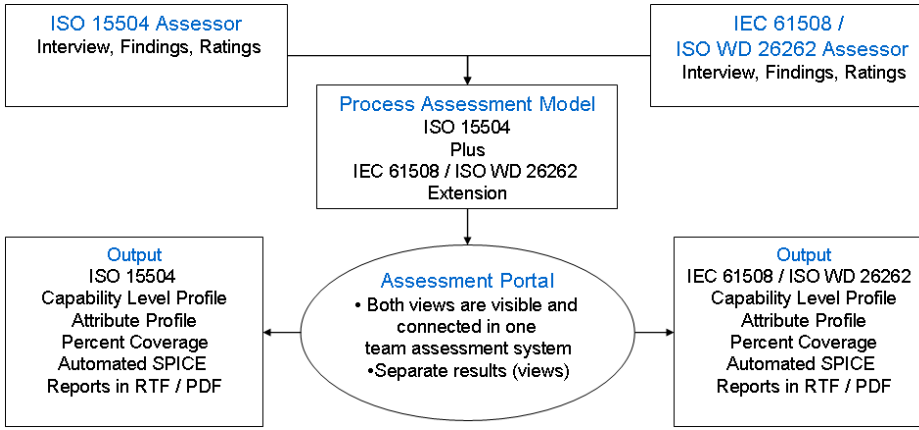


Figure 5: Integrated Team Assessment Process

5. *Assessment Systems and Results*

When performing an assessment you can switch between the Automotive SPICE and the Functional Safety view.

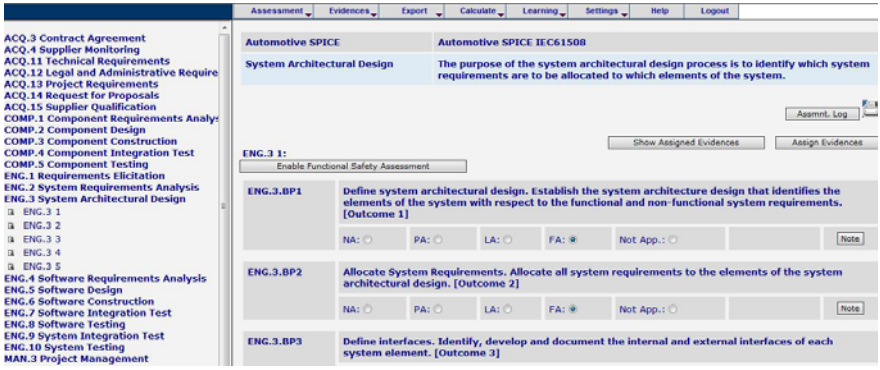


Figure 6: Automotive SPICE View – Selected Process System Architectural Design

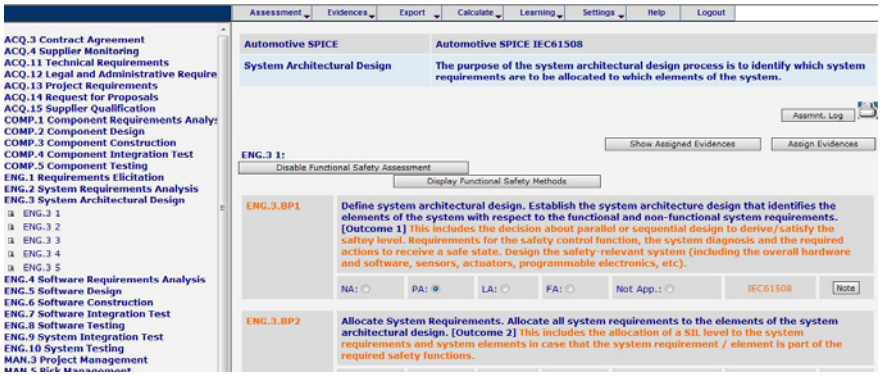


Figure 7: Extended Safety SPICE View – Selected Process System Architectural Design

In the functional safety view

Existing base practices have enhanced and include more safety relevant requirements.
Additional safety related practices have been added.
Functional safety methods can be displayed which assessors have to consider when assessing the use of specific methods.

The rating in the SPICE view and the Functional Safety view are stored separately. In most cases where a base practice has been extended it is more difficult to achieve the base practice when considering the additional safety aspects. Thus it made sense to store both ratings separately.

Table 5.1 Methods and measures for specifying software safety requirements		ASIL			
		A	B	C	D
1	Specification of software safety requirements in natural language	++	++	++	++
2a	Informal notations for requirements specification	++	++	+	+
2b	Semi-formal notations for requirements specification	+	++	++	++
2c	Formal notations for requirements specification	+	+	++	++
NOTE 1: The software safety requirements need to be described completely and consistently by an appropriate combination of methods 1 and 2x). NOTE 2: In case of model-based development, other functions according to 5.4.7 are to be described with the same combination of methods and measures used for the software safety requirements specification. NOTE 3: Computer-aided tools requirements specification should be applied according to WD° 26262-8, table 5.1.					

Table 5.2 Methods and measures for verifying software safety requirements		ASIL			
		A	B	C	D
1a	Informal verification	see next table			
1b	Semi-formal verification	+	+	++	++
1c	Formal verification	+	+	++	++

Figure 8: Extended Safety SPICE View – Functional Safety Methods – Sample Extraction

The assessment plan must consider a specific SIL level. Depending on the SIL level different selections of functional safety methods shall be considered from the underlying method tables.

To understand the functional safety methods related tables we explain how to interpret the lines (compare with Figure 8).

Interpretation of the first line in table 5.1 in Figure 8:

Specifications describing software requirements are highly recommended for all ASIL levels.

Interpretation of the lines 2a to 2c in table 5.1 in Figure 8:

For ASIL C a semi-formal or formal notation of requirements specification is necessary. A formal notation would mean to use a specification language / structure which allows to track and verify the completeness of requirements.

In Automotive SPICE the generic practices on level 2 and 3 (capability indicators) are being interpreted per process and for each process a capability level is determined. All generic practices on capability level 2 had to be extended in the Functional Safety view.

Generic Practice	Description
2.1.1	GPI 2.1.1 Identify the objectives for the performance of the process. ….. This includes objectives related with the coverage of system safety requirements in the system design.
2.1.2	GPI 2.1.2 Plan and monitor the performance of the process to fulfil the identified objectives. ….. The techniques and measures which are necessary during the safety lifecycle phase related with system design to achieve the safety integrity level are specified and planned.
2.1.3	GPI 2.1.3 Adjust the performance of the process. Process performance issues are identified. ….. Use of failure /error and probability metrics to measure the achievement of the saftey objectives. This includes the control of the performance of the safety life cycle phase related with customer requiremnts issues. Document the results and state of the verification or the reason for the failures where the customer requitrements have passed the verification.
2.1.4	GPI 2.1.4 Define responsibilities and authorities for performing the process. ….. This includes the definition of a role responsible for the safety requirements coverage in the dsystem design.
2.1.5	GPI 2.1.5 Identify and make available resources to perform the process according to plan. ….. This includes the selected set of safety methods and tools.
2.1.6	GPI 2.1.6 Manage the interfaces between involved parties. ….. The communication with the safety manager and safety responsible on the project level must be managed.
2.2.1	GPI 2.2.1 Define the requirements for the work products. The requirements for the work products to be produced are defined ….. This includes a documentation guidance about how to design safety systems and cover safety requirements. Justify and document the techniques and measures chosen to form an integrated set which satisfies the required safety integrity level.
2.2.2	GPI 2.2.2 Define the requirements for documentation and control of the work products. ….. This includes a full documentation of the safety concept and design and the classification of safety critical function flows inside the system design.The system design documentation shall be version

	controlled in a configuration management plan / system. Specify and document those techniques and measures which are necessary during the safety lifecycle phases to achieve the safety integrity level.
2.2.3	GPI 2.2.3 Identify, document and control the work products…... A professional traceable system is used for the safety documentation.
2.2.4	GPI 2.2.4 Review and adjust work products to meet the defined requirements. ….. Tracking of an detected problem /error to its root cause. (Reference in FMEA). This includes a review of the coverage of all specified safety requirements in the architectural design (safety target coverage, diagnostic coverage, etc.). Review the requirements for the safety-relevant software and hardware to ensure that they are adequately specified.

Figure 9: Extensions of Generic Practices on Capability Level 2

Figure 10: Attribute Rating Profile Example – Safety View Enabled

Figure 11: Attribute Rating Profile Example – Automotive SPICE (Compare with Figure 10)

Safety SPICE adds to the existing Automotive SPICE scope safety relevant practices, work products and methods. Thus a rating in Figure 11 will look different when adding the safety scope like in Figure 10.
The assessment portals used in the trial partnership allows to use and store (and compare) both views in one assessment.

Based on the ratings given by assessors in the Automotive SPICE and/or Functional Safety view reports are generated illustrating deviations and improvement potentials.

Base Practice	Description	Rate	Notes
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 safety 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).	P	No independent control function was included in the design. However, at certain check points the correctness of calculations is verified using diagnose modules.
..	..	..	..

Figure 12: Automatically Generated Reporting

6. Outlook and Future Usage

An Automotive SPICE assessment takes 3 to 4 days and includes a lot of effort for interviews and analysis work. An additional safety assessment with an additional set of processes would increase the number of days spent on assessments for projects. Thus the partnership mapped both models and created an integrated assessment approach. This will help saving the overall assessment time when applying both models.

It will also help to better integrate improvement plans in companies to match both goals, to achieve an Automotive SPICE level 3 and to achieve a specific ASIL level.

The results of this work will be made public in autumn 2008.

Currently trials using the integrated approach are running in major supplier companies. We have to thanks especially Continental Automotive and all other SOQRATES members (www.soqrates.org) for their contribution to the mapping.

7. References

[1] R. Messnarz, et. al, Assessment Based Learning Centers, in : Proceedings of the EuroSPI 2006 Conference, Joensuu, Finland, Oct 2006, also published in Wiley SPIP Journal (Software Process Improvement in Practice), Volume 12 Issue 6 , Pages 505 – 610, November/December 2007

[2] R. Messnarz, et.al, Better Software Practice for Business Benefit, IEEE Computer Society Press, 1999, Washington, Tokyo, Berlin

[3] R. Messnarz, From process improvement to learning organisations (p 287-294), Wiley SPIP Journal (Software Process Improvement in Practice), Volume 11 Issue 3 , Pages 213 - 335 (May/June 2006)

[4] Automotive SPICE, www.automotivespice.com

[5] Draft publications of ISO WD 26262, Road vehicles — Functional safety, Parts 1 – 8

[6] SOQRATES Initiative, www.sowrates.de

[7] HIS, www.his-automotive.de

MSc Leo Ross

Since autumn 2004, Hans-Leo Ross is in the Quality Business Process responsible for the implementation of the necessary measures for Functional Safety within Continental Automotive. Further more the general conformity of Engineering Processes to relevant standards (ASPICE, ISO TS 16949 etc.) as well as the coordination of the Conti Automotive Safety Manager belongs to the scope of responsibility.

Safety Managers in Conti Automotive are responsible for safety reviews and assessments in product area like chassis, brake, power train and interior electronic equipment.

In FAKRA (now Normenausschuss Automobil, German Automotive Standardization Body within VDA) and in the international standardization group ISO SC3/TC22/WG 16, Hans-Leo Ross is a nominated expert fort he coming standard ISO 26262.

At the University UNI-GH Paderborn, Hans-Leo Ross passed the Dipl.-Ing. In Communication and Electronic. During the years 1999 to 2003, Hans-Leo Ross graduated Master in Economy and Marketing and the Master of Advance Studies at the University of Basel (Switzerland).

The professional carrier passes System Engineering for Safety Concepts in the Oil and Gas industry as well as sales and Head of Product Management for Safety related control systems for plant automation and machinery applications. During this work, Hans-Leo Ross was frequently involved also in the development of safety standards such as IEC 61508.

Dr Richard Messnarz

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

PICO - Process Improvement Combined Approach 1995 - 1998,

Bestregit - Best Regional Technology Transfer, 1996 - 1999,

TEAMWORK - Strategic E-Working Platform Development and Trial, 2001-2002,

MediaISF – E-Working of media organisation for strategic collaboration on EU integration, 2001-2002

ORGANIC – EU Innovation Manager, 2003 – 2006

European Quality Network, 2005 – 2007

European Certificates Association, 2008 - 2010

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 principal ISO 15504 and Automotive SPICE assessor. He has worked as a consultant for many automotive firms, such as BOSCH, ZF TE, ZF N, ZF SACHS, Continental TEMIC, T-Systems, Magna Powertrain, Giesecke & Devrient, etc. in the last 18 years. 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).

MSc Stephan Habel

Since early 2008, Stephan Habel has held the position Functional SafetyManager within the Business Unit Transmisson. Besides, the general conformity of Engineering Processes to relevant standards (ASPICE, ISO TS 16949 etc.) as well as the coordination of the Project Safety Coordinator also belongs to his scope of responsibility.

Safety Managers at Conti Automotive are responsible for safety reviews and assessments in product areas such as Chassis, Brake, Powertrain and Interior Electronic Equipment.

He has been leading a team of electronic experts responsible for moderation of Hazard Analyze, FME(D)A and FTA.

From 1984 to 1989 Stephan Habel attended the Georg-Simon-Ohm University of Applied Sciences in Nuremberg and graduated as a Dipl.-Ing. (FH) in Electronics .

Since 1989 Stephan Habel has worked for Continental Temic. Starting out in a test center, he moved to the development department in 1993 where he was frequently involved in System Engineering Powertrain and Chassis Application for the automotive industry.

In 2003 Stephan Habel was Conti Temic's representative in the "first" Soqurates Group dealing with SPICE.

MSc Jürgen Unterreitmayer

Jürgen Unterreitmayer is an experienced test manager and project manager and participated in many development and testing projects in the telecommunications, automotive, logistics and transport industry. His main activity over the last years was the assessment and improvement of software development and testing processes.

Before specialising in software test and quality assurance Jürgen held various positions at the European Space Agency, Siemens, Viag Interkom and Sun Microsystems, mainly in hardware / software development, product management and business development.

Since October 2007 he is working as Delivery Manager for the Industry & Engineering business unit of SQS AG, Europe`s biggest independent consulting company focused on software quality management and assurance. Delivery Managers' responsibilities at SQS AG include team management, project management and consulting on senior level.

Before that he was head of the branch office in Munich for imbus AG, a company specialised mainly in soft-ware testing, also responsible for leading a team of consultants.

Jürgen passed the Dipl. Ing. in electronic engineering at the Technical University of Munich.

Jürgen is certified as ISTQB Certified Tester and SPICE Assessor. Based on his experience he is executing trainings and supporting customers with coaching, mainly in Test Management.

A Software Engineering Lifecycle Standard for Very Small Enterprises

Claude Y. Laporte[1], Simon Alexandre[2], and Rory V. O'Connor [3]

[1] École de technologie supérieure, Montréal, Canada
[2] Centre d'Excellence en Technologies de l'Information et de la Communication, Charleroi, Belgium
[3] School of Computing, Dublin City University, Dublin, Ireland
Claude.Y.Laporte@etsmtl.ca, simon.alexandre@cetic.be, roconnor@computing.dcu.ie

Abstract. Industry recognizes that very small enterprises (VSE), that develop parts having software components, are very important to the economy. These parts are often integrated in products of larger enterprises. Failure to deliver on time, within budget a quality product threatens the competitiveness of both organizations. One way to mitigate these risks is by having all suppliers of a product chain to put in place recognized engineering practices. Many international standards and models like ISO/IEC12207 or CMMI have been developed to capture proven engineering practices. However, these standards were not designed for very small development organizations, those with less than 25 employees, and are consequently difficult to apply in such settings. An ISO/IEC JTC1/SC7[1] Working Group has been established to address these difficulties by producing a tailored software engineering standard to VSE.

Keywords: ISO, Lifecycles, Very Small Enterprises, Standards

1 Introduction

Today the ability of organizations to compete, adapt, and survive depend increasingly on software. By 2010, it is estimated that cellular phone will contain 20 million lines of code, an automobile manufacturer estimated that its cars will have up to 100 million lines of code [1]. These manufacturers depend increasingly on the components produced by their suppliers. A manufacturing chain, of large mass market products, often has a pyramidal structure. The pyramid is composed, of a layer of dozens of main suppliers which are supplied by a layer of hundreds of smaller suppliers. These small suppliers layer have thousands of very small suppliers. As an example, a large mass product manufacturer integrated in one of its product a part, with an unknown software error, produced by one of its 6000 producers[2]. The defective part resulted in a multi-million dollar loss by the manufacturer. The need for international software engineering standards is clear.

There is evidence that the majority of small software organizations are not adopting existing standards as they perceive them as being orientated towards large organizations. Studies have shown that small firms' negative perceptions of process model standards are primarily driven by negative views of cost, documentation and bureaucracy. In addition, it has been reported that VSEs find it difficult to relate ISO/IEC 12207 to their business needs and to justify the application of the international standards in their operations. Most VSEs cannot afford the resources for, or see a net benefit in, establishing software processes as defined by current standards (e.g. ISO/IEC 12207) and maturity models such as the Capability Maturity Model Integration CMMI) developed by the Software Engineering Institute [3].

Accordingly there is a need to help these organizations understand and use the concepts, processes and practices proposed in the ISO/IEC JTC1/SC7's international software engineering standards. This paper presents a new project intended to facilitate access to, and utilization of, ISO/IEC JTC1/SC7 software engineering standards in very small enterprises.

This paper is divided into six sections. Section 2 presents the concept of a VSE and describes the characteristics that distinguish a VSE from other organizations. Section 3 presents a historical perspective on the

[1] ISO/IEC JTC 1/SC7 stands for the International Organization for Standardization/ International Electrotechnical Commission Joint Technical Committee 1/Sub Committee 7, which is in charge of the development and maintenance of software and systems engineering standards

events that led to an ISO/IEC JTC1 SC7 project proposal for VSEs and section 4 presents the results of a survey that was developed to question VSEs about their utilization of ISO/SC7 standards. Section 5 explains the approach being take by the VSE working group and finally section 6 presents concluding remarks and discusses future actions.

2 Very Small Enterprises

The definition of "*Small*" and "*Very Small*" Enterprises is challengingly ambiguous, as there is no commonly accepted definition of the terms. For example, the participants of the 1995 CMM tailoring workshop [4] could not even agree on what "*small*" really meant. Subsequently in 1998 SEPG conference panel on the CMM and small projects [5], small was defined as "*3-4 months in duration with 5 or fewer staff*". Johnson and Brodman [6] define a small organization as "*fewer than 50 software developers and a small project as fewer than 20 software developers*". Another definition for VSE introduced by Laporte et al [7] as "*any IT services, organizations and projects with between 1 and 25 employees*".

To take a legalistic perspective the European Commission [8] defines three levels of small to medium-sized enterprise (SME) as being: **Small to medium** - "*employ fewer than 250 persons and which have an annual turnover not exceeding 50 million Euro, and/or an annual balance sheet total not exceeding 43 million Euro*"; **Small** - "*which employ fewer than 50 persons, and whose annual turnover or annual balance sheet total does not exceed 10 million Euro*" and **Micro** - "*which employ fewer than 10 persons and whose annual turnover*".

To better understand the dichotomy between the definitions above it is necessary to examine the size of software companies operating in the market today. In Europe, for instance, 85% of the Information Technology (IT) sector's companies have 1 to 10 employees[2]. In the context of indigenous Irish software firms 1.9% (10 companies), out of a total of 630 employed more than 100 people whilst 61% of the total employed 10 or fewer, with the average size of indigenous Irish software firms being about 16 employees [9]. In Canada, the Montreal area was surveyed, it was found that 78% of software development enterprises have less than 25 employees and 50% have fewer than 10 employees [10]. In Brazil, small IT companies represent about 70% of the total number of companies [11].

Therefore based on the above discussions, for the purposes of this paper we are adopting the definition for VSE introduced in [7] as "*any IT services, organizations and projects with between 1 and 25 employees*".

2.1 Characteristics of a VSE

The unique characteristics of small entrepreneurial businesses as well as the uniqueness of their situations of necessity make their style of business different [12]. Some of the unique differences between very small and large businesses behavior are given in Table 1.

Table 1. Characteristic differences between large firms and small firms.

Characteristic	Small firm	Large firm
Planning orientation	Unstructured/operational	Structured/strategic
Flexibility	High	Structured/strategic
Risk orientation	High	Medium
Managerial process	Informal	Low
Learning and knowledge absorption capacity	Limited	High
Impact of negative market effects	More profound	More manageable
Competitive advantage	Human capital centered	Organizational capital centered

Software VSEs are subject to a number of distinctive and intrinsic characteristics that make them different from their larger counterparts, therefore affecting the contents, the nature and the extent of the activities. We classify VSE characteristics according to four main categories: Finance, Customer, Internal Business Processes and Learning and Growth:

VSEs are economically vulnerable as they are driven by cash-flow and depend on project profits, so they need to perform the projects within the budget. They tend to have low budgets which have many impacts, such

[2] http://www.esi.es/en/main/iitmark.html

as: Lack of funds to perform corrective post delivery maintenance; Few resources allocated for training; Little or no budget to perform quality assurance activities; No budget for software reuse processes; Low budget to respond to risks; and Limited budget to perform Process Improvement and /or obtain a certification/assessment.

Typically the VSEs product has a single customer, where the customer is in charge of the management of the system; the software integration, installation and operation. It is normal practice for the customer to not define quantitative quality requirements and for customer satisfaction depends on the fulfillment of specific requirements that may change during the project. A close relationship between all involved project members including the customer shows that software development in small and very small companies is strongly human oriented and communication between them is important. For example, in contrast to small companies, very small companies often do not have regularly project meetings [13].

The internal business process of VSEs are usually focused on developing custom software systems, where the software product is elaborated progressively and which typically does not have strong relationship with other projects. Typically most management processes (such as human resource and infrastructure management) are performed through informal mechanisms, with the majority of communication, decision making and problem resolution being performed face to face.

The learning and growth characteristics of VSE are typified by a lack of knowledge (or acceptance) of software process assessment and improvement and a lack of human resources to engage in standardization. It is usual for a negative perception of standards as being made by large enterprises, for large enterprises [9].

3 History of the ISO/IEC Working Group for VSEs

The mandate of ISO/IEC JTC1/SC7 is the standardization of processes, supporting tools and supporting technologies for the engineering of software products and systems. A description of SC7 and of the development of ISO/IEC JTC1/SC7 standards is presented in [14]. In this section, a brief history of the events leading to the creation of a new ISO/IEC JTC1/SC7 Working Group (WG) is presented. A detailed description of its history is available in [3].

At the May 2004 SC7 Plenary meeting in Brisbane, Canada raised the issue of small enterprises requiring standards adapted to their size and maturity level. The current software engineering standards target (or are perceived as targeting) large organizations. A meeting of interested parties was organized and a consensus was reached on general objectives for a future working group:

- To make the current software engineering standards more accessible to VSEs;
- To provide documentation requiring minimal tailoring and adaptation effort;
- To provide harmonized documentation integrating available standards:
  - Process standards
  - Work products and deliverables
  - Assessment and quality
  - Modeling and tools
- To align profiles, if desirable, with the notions of maturity levels presented in ISO/IEC 15504.

In March 2005, the Thailand Industrial Standards Institute (TISI) invited a Special Working Group (SWG) to advance the work items defined at the Brisbane meeting. A key topic of discussion was to clearly define the size of VSE that the SWG would target, consensus being reached on IT services, organizations and projects with 1 to 25 employees. The major output of this one-week meeting was a draft of the New Work Item (NWI) to be tabled at the next SC7 meeting.

In May 2005, a resolution was approved to distribute for ballot the NWI Proposal for the development of Software Life Cycle Profiles and Guidelines for use in Very Small Enterprises. Twelve countries voted in favor of the NWI Proposal [15]. As a result of this vote, the Project was approved and the new working group, WG24, was established.

The Thailand Industrial Standards Institute (TISI) sent out a second invitation to participate in the SWG, to be held in September 2005 in Bangkok. The main objective of the meeting was to prepare material that would be presented to WG24 in order to facilitate the start-up of the working group that was scheduled for October 2005 in Italy.

In October 2005, Italy hosted ISO/IEC JTC1 SC7 Interim Meeting. The New Work Item was updated in order to take into account relevant comments received during balloting, and the requirements were validated by WG members. In addition, some VSE Business Models were identified, as was a strategy for creating profiles. Finally, WG24 decided to conduct a survey to collect relevant information from VSEs around the world.

4 Gathering VSE Requirements

In 1997, the Technical Council on Software Engineering responsible for the IEEE Software Engineering Standards conducted a survey to capture information from software engineering standards users in order to improve those standards [16]. They gathered 148 answers, mainly from the USA (79%) and large companies (87% of them having more than 100 employees). The main application domains of the survey respondents were IT (22%), military (15%) and aerospace (11%). (It should be noted that the purpose of this section is not to systematically compare the two sets of survey results). Even though the IEEE survey objectives differ from those of the ISO/IEC survey, there are some interesting common findings. In response to the question concerning the reasons why their organization does not use standards, 37% said that the standards were not available in their facilities, while 37% explained that they use other standards. In fact, the IEEE survey underscores the fact that ISO/IEC standards are often used in organizations, rather than the IEEE standards.
The IEEE survey underlined the difficulties regarding IEEE standards use reported by the respondents. The two main difficulties were a lack of understanding of the benefits (28%) and a lack of useful examples (25%). The survey also revealed how IEEE standards are used in organizations. Most of the organizations claimed to use IEEE standards for internal plan elaboration. The IEEE survey gathered several new requirements about IEEE standards being requested by the respondents. These were principally examples and templates of deliverables, support for metrics and measurement, help on life cycle process definition, a training course and support for small, rapid application development efforts.
The WG24 survey was developed to question VSEs about their utilization of ISO/SC7 standards and to collect data to identify problems and potential solutions to help them apply standards and become more competitive. From the very beginning, the working group drew up several working hypotheses regarding VSEs. The survey was intended to validate some of these hypotheses, such as the following:

- The VSE context requires light and well-focused life cycle profiles.
- Particular business contexts require particular profiles.
- There are significant differences, in terms of available resources and infrastructure, between a VSE employing 1 to 10 people and an Information Technology (IT) department of the same size in a larger company.
- VSEs are limited in both time and resources, which leads to a lack of understanding of how to use the standards for their benefit.
- Benefits for VSEs may include recognition through assessment or audit by an accredited body.

The survey questionnaire and an introductory text were developed by the WG24 and translated into 9 languages: English, French, German, Korean, Portuguese, Thai, Turkish, Russian and Spanish. The survey is made up of 20 questions structured in 5 parts: General information, Information about standards utilization in VSEs, Information about implementation and assessment problems in VSEs, Information about VSE needs and Information about justification for compliance to standard(s). Over 392 responses had been collected from 29 countries.

4.1 Categorization of the sample according to the size criterion

Of the 392 responders, 228 are enterprises with 0 to 25 employees (58%), as illustrated in Figure 1. Note that responders of small organizations (<25 persons) that are a part of a larger enterprise are not included in these 228 responses. These 228 VSEs constitute the sample for this study. The following paragraphs present findings common to the 228 VSEs and identifies correlations inside the sample, and findings that differ from those of the bigger companies that contributed to the survey.

This categorization and several studies underscore the differences between micro, small and medium enterprises in terms of available resources. Therefore, WG24 decided to focus on the first category (micro enterprises with 0-9 employees) and on a subpart of the small enterprise category (10-25 employees).

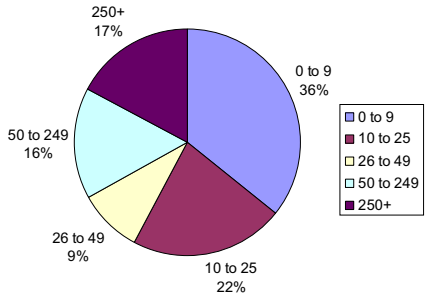


Fig. 1. Number of employees in the enterprises surveyed.

4.2 General characteristics

Here, we draw attention to some weaknesses of the sample itself. Since the survey was initiated through WG24 contacts without building a true random sample, the survey results may have been impacted. The first observation about the respondent sample, as illustrated in Table 2, is the geographical distribution of answers. We collected a high number of responses from Latin America (46%), mainly from Colombia and Brazil.

Table 2. Number of Survey Responses per Country.

Country	No. of Responses	Country	No. of Responses
Argentina	2	Italy	2
Australia	8	Japan	3
Belgium	10	Korea (South)	4
Brazil	68	Mexico	20
Bulgaria	3	New Zealand	1
Canada	8	Peru	4
Chile	1	Russia	4
Colombia	88	South Africa	10
Czech Rep.	3	Spain	2
Ecuador	9	Taiwan	1
Finland	13	Thailand	52
France	3	Turkey	1
India	57	United Kingdom	2
Ireland	10	United States	3

At the same time, we received only a few responses from European countries (48), Japan (3) and the United States (3). Therefore, our results might only generalize to the broader populations of projects in each region to the extent that this sample represents them. Moreover, we have no evidence that participating companies are representative of the situation in their own countries.

4.3 Use of standards

An interesting finding of the survey is the difference in the percentage of certified companies with regard to company size: less than 18% of VSEs are certified, while 53% of larger companies (more than 25 employees) claim to be certified. Furthermore, among the 18% not certified, 75% do not use standards. In larger companies using standards, two families of standards and models emerge from the list: ISO standards (55%) and models from the Software Engineering Institute (SEI) (47%).

WG24 anticipated the weak use of standards by VSEs by asking questions designed to provide a better understanding of the reasons for this. There are three main ones, as shown in Figure 2. The first is a lack of resources (28%); the second is that standards are not required (24%); and the third derives from the nature of the

standards themselves: 15% of the respondents consider that the standards are difficult and bureaucratic, and do not provide adequate guidance for use in a small business environment.

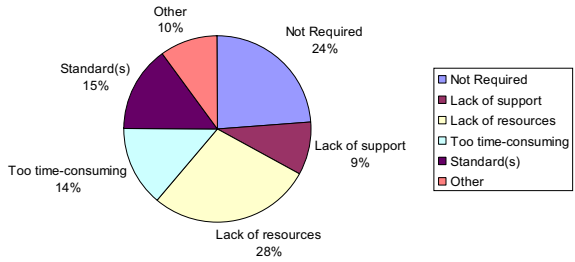


Fig. 2. Why VSEs do not use standards.

For a large majority (74%) of VSEs, it is very important to be evaluated or certified against a standard. ISO certification is requested by 40% of them. Of the 28% requesting official market recognition, only 4% are interested in a national certification. From the VSE perspective, some benefits provided by certification are:

- Increased competitiveness
- Greater customer confidence and satisfaction,
- Greater software product quality
- Increased sponsorship for process improvement
- Decreased development risk
- Facilitation of marketing (e.g. better image)
- Higher potential to export

However, VSEs are expressing the need for assistance in order to adopt and implement standards. Over 62% would like more guidance with examples, and 55% are asking for lightweight and easy-to-understand standards complete with templates. Finally, the respondents indicated that it has to be possible to implement standards with minimum cost, time and resources. All data about VSEs and standards clearly confirm WG24's hypothesis and the requirements. Therefore, WG24 uses this information to help define its approach for the development of profiles, guides and templates to meet VSE needs.

5 The WG24 Approach

The approach used by WG24 had to take into account, as a starting point, the ISO requirements in terms of standard definition. Indeed, since an international standard dedicated to software lifecycle was already available (i.e. ISO/IEC 12207) [17], WG24 had to use the concept of ISO profiles (ISP – International Standardized Profile) in order to develop the new standard for VSEs. A Profile is defined as "A set of one or more base standards and/or ISPs, and, where applicable, the identification of chosen classes, conforming subsets, options and parameters of those base standards, or ISPs necessary to accomplish a particular function" [18]. From a practical point of view, a Profile is a kind of matrix that identifies precisely all elements that are taken from existing standards from those that aren't.

The overall approach followed by WG24 to develop this new standard for VSE consisted in three steps:

- Select ISO/IEC12207 process subset applicable to VSEs of less than 10 employees
- Tailor the subset to fit VSE needs
- Develop guidelines

Firstly, since WG24 wished to prepare an initial set of software development standards as quickly as possible, WG24 analyzed international reference standards and models that could help subset ISO/IEC 12207 for low maturity VSEs. To achieve these initial products quickly, WG24 began a search for existing standards or models that could be tailored. Moprosoft, a Mexican standard developed to assist Mexican small and medium enterprises (SMEs) has been selected in order to achieve this objective [19].

Moprosoft uses ISO/IEC 12207 as a general framework. It borrows practices from ISO9001, the Capability Maturity Model Integration (CMMI) developed by the Software Engineering Institute, the Project Management Body of Knowledge (PMBOK) and the Software Engineering Body of Knowledge SWEBOK.

However, WG24 felt that Moprosoft was addressing the needs of organizations larger than targeted VSEs. Therefore, as a second step, WG24 decided to tailor Moprosoft in order to address key characteristics of low maturity VSEs. The tailoring approach lead to the development of incremental profile targeting as starting point, low maturity VSE of less than 10 employees and, in a second phase, those with 10 to 25 employees. Therefore, the first profile, developed by WG24, contains basic activities coming from project management and software development related processes. The idea was to concentrate on core activities that a low maturity VSE should perform.

The first document of the family of documents developed by WG24, titled Overview, introduces the major concepts required to understand and use the suite of documents. It introduces the business aspects, characteristics and requirements of VSEs, and clarifies the rationale for VSE-specific profiles, documents, standards and guides. It also introduces basic process, lifecycle and standardization concepts, and the 29110 family of documents. It is targeted both at a general audience interested in these documents, and more specifically at users of these documents. The Overview is identified as technical report (TR) TR 29110-1.

The second set of documents; titled Profiles are defined to formally package references to and/or part of other documents in order to adapt them to the VSEs needs and characteristics. Preparing profiles is an ISO/IEC JTC1 defined process. It involves producing two types of documents: a framework and taxonomy and a profile specification:

- **Framework and Taxonomy** - The Framework and Taxonomy document (ISP29110-2) establishes the logic behind the definition and application of profiles. It specifies the elements common to all profiles (structure, conformance, assessment) and introduces the taxonomy (catalogue) of 29110 profiles. It is targeted at authors and reviewers of ISPs, authors of other parts, and authors of other VSE-targeted profiles. The Framework and Taxonomy is applicable to all profiles and identified as TR 29110-2
- **Profile Specifications** - There is a profile specification document for each profile. Its purpose is to provide the definitive composition of a profile, provide normative links to the normative subset of standards (e.g. ISO/IEC 12207) used in profile, and provide informative links (references) to "input" documents (e.g. 90003, SWEBOK, PMI). It is targeted at authors/providers of guides, and authors providers of tools and other support material. There is one profile specification document for each profile, identified as 29110-4.x, where x is the number assigned to the profile.

The third set of documents, titled Guides, contain implementation guidelines (domain specific) on how to perform the processes to achieve the maturity levels (e.g. recommended activities, measures, techniques, templates, models, methods ...). Guides are developed for the process implementation and for the assessment based on the domain's issues, business practices and risks. Guides are targeted at VSE, and should be VSE accessible, both in terms of style and cost. There are two guides: an assessment guide and a management and engineering guide:

- **Assessment Guide** - This guide describes the process to follow to perform an assessment to determinate the process capabilities and the organizational process maturity. This is, when an organization wants an assessment execution in order to obtain a process capability profile of the implemented processes and an organizational process maturity level. It is also applicable to the situation where customer asks for a third-party assessment execution in order to obtain a capability level profile of the implemented process by the software development and maintenance provider. It is also suitable for self-assessment. The Assessment Guide is applicable to all profiles and identified as TR 29110-3
- **Management and Engineering Guides** - The management and engineering guides provide guidance on its implementation and use or a profile. It is targeted at VSE (management and technical staff), VSE-related organizations (technology transfer centers, government industry ministries, national standards, consortiums and associations, academic use for training, authors of derived products (software, courseware, and acquirer and suppliers. There is one management and engineering guide document for each profile, identified as 29110-5.x, where x is the number assigned to the profile. This number matches the number assigned to the profile specification.

The third step of the approach consisted in defining guidelines explaining in more details the processes defined in the profile. These guidelines will be published as ISO Technical Reports which should be freely accessible to VSEs. These guidelines integrate a series of deployment packages. A deployment package is a set of artifacts developed to facilitate the implementation of a set of practices, of the selected framework, in a VSE. But, a deployment package is not a process reference model. The elements of a typical deployment package are: process description (e.g. activities, inputs, outputs, and roles), guide, template, checklist, example, presentation material, reference and mapping to standards and models, and a list of tools. Packages are designed such that a VSE can implement its content, without having to implement the complete framework at the same time. The first four deployment packages being developed are: requirements analysis and management, change

management, testing and project management. Future deployment packages are: architecture, issue tracking, unit testing and coding. The table of content of a deployment package is illustrated in table 3.

Table 3. Table of Content of a deployment package.

1. Introduction
Purpose of this document
Key Definitions
2. Why this Process is important
3. Overview of Main Tasks
3.1 Tasks
3.2 Roles and artifacts
3.3 Activity Lifecycle and examples of lifecycles
Annex A Templates
Annex B Checklists
Annex C Coverage Matrices (ISO 12207, ISO 9001, CMMI)
Annex D Tools
Annex E Training Material
Annex F Deployment Package Evaluation Form

5.1 Recent Developments

At the Montreal meeting of WG24, in October 2007, the requirement analysis and management deployment package has been reviewed and received a broad support from the group members. The group decided to develop following deployment packages for its next meeting in Berlin: configuration management, project management, and testing.

Having profiles and guides for VSEs is not sufficient to ensure broad utilization and adoption: they have to be tested with real VSEs of a few countries. The Mexican delegation presented the result of the introduction, as pilot projects, of the first profile developed by WG24, in Latin American countries [20]. Also a new country, Columbia, and a new organization, the European Software Institute (ESI), joined WG24.

6 Conclusion and Future Work

Industry recognizes the value of VSEs in their contribution of valuable products and services. About 75% of software enterprises worldwide have fewer than 25 employees. ISO/IEC JTC1 SC7 standards are not easily applied in VSEs that generally find standards difficult to understand. Hence, VSEs require further guidance in order to integrate standards into their practices. ISO/IEC JTC1 SC7 decided to establish a new working group to address these issues.

With regard to future work, WG24 plan to invite VSEs to participate in the field trials before the standards get published by ISO Since a few WG24 delegates are already working closely with VSEs, they will play a key role in the coordination of the trials. Trials will help validate the approach and obtain feedback in order to improve the documents before going for ISO/IEC publication. WG24 is planning to produce a Final Draft in 2009. Publication by ISO/IEC is scheduled for 2010. In the meantime, deployment packages will be made available, to VSEs, on public web sites.

Additional Information
The following Web sites provide more information as well as articles and eventually deployment packages, which members of WG24 will develop:
http://profs.logti.etsmtl.ca/claporte/English/VSE/index.html
http://www.cetic.be/indexEN.php3

References

1. Charette, R.N., Why Software Fails, Spectrum, IEEE Computer Society, September 2005, pp 42-49
2. Shintani, K, Empowered Engineers are Key Players in Process Improvement, Presentation at the First International Research Workshop for Process Improvement in Small Settings, Software Engineering Institute, CMU/SEI-2006-SR-01, Pittsburgh, PA, 2006

3. Laporte, C.Y.; April, A.; Applying Software Engineering Standards in Small Settings: Recent Historical Perspectives and Initial Achievements. In: Proceedings of the First International Research Workshop for Process Improvement in Small Settings. Software Engineering Institute, Carnegie Mellon University, CMU/SEI-2006-Special Report-001, January 2006, pp. 39-51.
4. Ginsberg, M.; Quinn, L.; Process Tailoring and the Software Capability Maturity Model, Software Engineering Institute, CMU/SEI-94-TR-024, November 1995.
5. Hadden, R., Key Practices to the CMM: Inappropriate for Small Projects, Panel, Proceedings of the Software Engineering Process Group Conference, Chicago, 1998.
6. Johnson, D.; Brodman, J.; Applying the CMM to Small Organizations and Small Projects, Proceedings of Software Engineering Process Group Conference, Chicago, 1998
7. Laporte, C.Y.; April, A. and Renault, A.; Applying ISO/IEC Software Engineering Standards in Small Settings: Historical Perspectives and Initial Achievements, Proceedings of SPICE Conference, Luxembourg, 2006.
8. European Commission, 2005, The New SME Definition: User Guide and Model Declaration, available at: http://europa.eu.int/comm/enterprise/enterprise_policy/sme_definition/sme_user_guide.pdf
9. Coleman, G.; O'Connor, R.; Investigating Software Process in Practice: A Grounded Theory Perspective, Journal of Systems and Software, Vol. 81, No. 5, pp 772-784, 2008.
10. Laporte, C.Y.; Renault, A.; Desharnais, J. M.; Habra, N.; Abou El Fattah, M.; Bamba, J. C.; Initiating Software Process Improvement in Small Enterprises: Experiment with Micro-Evaluation Framework, SWDC-REK, International Conference on Software Development, University of Iceland, Reykjavik, Iceland, May 27-June 1, 2005, pp. 153-163.
11 Anacleto, A.; von Wangenheim, C.G.; Salviano, C.F.; Savi, R.; Experiences gained from applying ISO/IEC 15504 to small software companies in Brazil, 4th International SPICE Conference on Process Assessment and Improvement, Lisbon, Portugal, April 2004.
12. Mtigwe, B., The entrepreneurial firm internationalization process in the Southern African context: A comparative approach, International Journal of Entrepreneurial Behavior & Research, Vol. 11 No. 5, 2005, pp. 358-377
13. Hofer, C.; Software Development in Austria: Results of an Empirical Study among Small and Very Small Enterprises, Proceedings of the 28th Euromicro Conference, 2002: 361-366
14. Coallier, F.; International Standardization in Software and Systems Engineering, Crosstalk, February 2003, pp. 18-22.
15. New Work Item Proposal – Software Life Cycles for Very Small Enterprises, ISO/IEC JTC1/SC7 N3288, May 2005. http://www.jtc1-sc7.org/.
16. Land, S. K. (1997). Results of the IEEE Survey of Software Engineering Standards Users. Software Engineering Standards Symposium and Forum, 1997. Emerging International Standards. ISESS 97, Walnut Creek, CA, June 1-6, pp. 242 – 270.
17. ISO/IEC 12207:2008, Information technology – Software life cycle processes. International Organization for Standardization/International Electrotechnical Commission: Geneva, Switzerland.
18. ISO/IEC TR 10000-1:1998, Information technology: Framework and taxonomy of International Standardized Profiles Part 1: General principles & documentation framework.
19. NMX-059-NYCE-2005, Information Technology-Software-Models of Processes and Assessment for Software Development and Maintenance. Part 01: Definition of Concepts and Products; Part 02: Process Requirements (MoProSoft); Part 03: Guidelines for Process Implementation; Part 04: Guidelines for Process Assessment (EvalProSoft), Ministry of Economy, Mexico, 2005.
20. Oktaba, H., Felix G., Mario P., Francisco R., Francisco P. and Claudia, A.; Software Process Improvement: The Competisoft Project, IEEE Computer, October 2007, Vol. 40, No 10

Comparison of CMMI-SVC and ISO20000 – a case study

Lic. Tech. Risto Nevalainen *(risto.nevalainen@fisma.fi)*
Finnish Software Measurement Association FiSMA ry
Espoo, Finland

M.Sc. (Eng) Mika Johansson *(mika.johansson@spinet.fi)*
Spinet Oy
Espoo, Finland

Abstract

Service is a growing business in IT industry. Software development is already global and can be outsourced to low-cost countries. Maybe that is possible also for services in future, but nowadays it is mainly local business. It requires close, continuous communication with customer and fast response to urgent service needs. Most major IT companies in Europe are investing in service business, because it is the main growth area in their business portfolio.

Most well known models in IT service management are ITIL (IT Infrastructure Library) and ISO/IEC 20000. Current version of ITIL is 3.0. It is mainly a collection of best practices to implement good service management processes. ISO/IEC 20000 is a set of requirements, which can be seen as a reference model and a certification scheme for IT services. Also CMMI family is extending with a new model, CMMI for Services (CMMI-SVC).

Many IT organisations use already some model for software and systems engineering. IT service management models are still under intensive development and frequent change, because they are not so well established as engineering models. So, most organisations have to select their initial IT service model in this dynamic situation.

Finnish Software Measurement Association FiSMA is a national network in Finland supporting members in adopting new concepts and standards. We did a detailed mapping between ISO/IEC 20000 and CMMI-SVC to highlight model similarities and differences. We run also several pilots in one of our member company, TietoEnator.

This article describes piloting of CMMI-SVC and ISO/IEC 20000 in TietoEnator. Pilot was done in two TE units during spring 2008. Both piloting units were located in Finland, and their business is either local or Nordic. The ultimate goal of pilots was to find synergies between two approaches, and allow an effective implementation of both CMMI-SVC and ISO/IEC 20000.

Keywords

CMMI-SVC, ISO/IEC 20000, ITIL, Service Management

1 Introduction

Software development has been seen as a set of processes and models for a long time. Some software engineering concepts like waterfall and incremental development have existed for centuries. Also assessment models for software develoment are well established. First precedessors of CMMI were published in late 1980´s and SPICE during early 1990´s. We can say, that their current versions are already a second generation of assessment models. New innovations, like agile development and XP, are changing software development to be more effective and customer focused. Certification of software companies has been based mainly on ISO9001, and in some sectors also on CMMI or SPICE.

IT service management processes are less mature. The main process model is ITIL (IT Infrastructure Library). First version of ITIL was published in late 1980´s, but it became never so popular as CMMI or SPICE. Current version of ITIL is 3.0. First standards for service management have been published in early 2000´s. Primary standard ISO/IEC 20000 (IT Service Management) was published in 2005. It can be used both as a guidance for implementation or as a reference model for certification. Some countries have started certification programs in IT service business, based on ISO/IEC 20000.

IT service companies and businesses are facing a decision problem: What model or standard we should adopt as an implementation and assessment framework? Which approach is best for improvement of our IT service processes? How the future of standards and models looks like? We know already that most popular IT service management models are either changing or are still in a piloting phase. New version of ISO/IEC 15504 as a whole is under evaluation, due to ISO rules to renew standards in 5-year cycle. CMMI for Services (CMMI-SVC) is still under development and may be published in first quarter of 2009. ISO/IEC 20000 is also under major development, to be compliant with ITIL 3.0 and a number of ISO standards. Even ISO9000 series is changing and new ISO9001:2008 will be published October 2008. IT Governance brings new concepts and ideas in IT services.

Finnish Software Measurement Association (FiSMA) has participated actively in many major process improvement initiatives during last two decades. Our member companies are active in adopting new innovations. One of FiSMA services for members is piloting of new ideas and models.

TietoEnator (later also shortly TE) is the largest member of FiSMA. They need several approaches because of their different business logics and customer requests. TE has used CMMI-DEV model in internal assessment of software development since 2004. CMMI-SVC is an interesting option for them as an easy and reusable extention of CMMI-DEV, mainly in application management and customer support businesses. TE has also large-scale operation management business, which is largely compliant with ITIL model. Some TE customers may require certification of service management and/or operations management processes. Then ISO/IEC 20000 is then an interesting option, and almost the only choice.

2 CMMI-SVC and ISO/IEC 20000 models

2.1 CMMI for Services (CMMI-SVC)

The work with CMMI-SVC model started in 2005. First drafts were opened for public review and commenting in early 2006. Major delay in publication was caused when DoD, the main sponsor of SEI, wanted more discussion and consensus about the model and CMMI architecture in general. Interesting is to see, that Garnegie Mellon University has its own service management model eSCM for customers and suppliers of service sourcing.

CMMI-SVC model is still a draft version. Both pilots were done by using version 0.5. SEI has started a large project to finalise the model and publish it in March 2009 as the first official version.

CMMI architecture is based on three constellations (Development, Service, Acquisition). The core part of these constellations is same, including 16 process areas. All constellations have some additional

process areas like “plug-ins”. This modular approach allows reuse of process investments and makes adoption of CMMI-SVC easier for CMMI-DEV users.

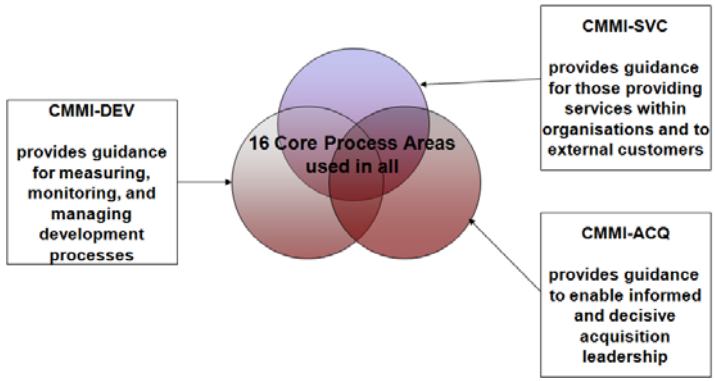


Figure 1. CMMI constellations and core process areas

CMMI-SVC has nine additional process areas to core 16, including together 25 processes. Six process areas are the core set of service specific processes and rest three can be combined in 8 different ways.

Table 1. CMMI-SVC process areas. Service specific process areas are shown in *italics*. REQM process area has one additional goal in SVC constellation. ** = optional process in SVC model.*

Category	**Process Area**
Process Management	Organizational Innovation and Deployment (OID) Organizational Process Definition (OPD) Organizational Process Focus (OPF) Organizational Process Performance (OPP) *Organizational Service Management (OSM)** Organizational Training (OT)
Project Management	*Capacity and Availability Management (CAM)* Integrated Project Management (IPM) Project Monitoring and Control (PMC) Project Planning (PP) *Requirements Management (REQM)* Risk Management (RSKM) Quantitative Project Management (QPM) *Service Continuity (SCON)** Supplier Agreement Management (SAM)
Service Establishment and Delivery	*Incident and Request Management (IRM)* *Service Delivery (SD)* *Service System Development (SSD)** *Service Transition (ST)*
Support	Causal Analysis and Resolution (CAR) Configuration Management (CM) Decision Analysis and Resolution (DAR) Measurement and Analysis (MA) *Problem Management (PRM)* Process and Product Quality Assurance (PPQA)

2.2 ISO/IEC20000:2005 and FiSMA IT Service Management Assessment Model

The current published version of the ISO/IEC 20000 standard is from 2005. It consists of two parts. Part 1 (named here as ISO/IEC 20000-1) is the requirements part for certification purposes. Part 2 gives additional guidance and recommendations for providing IT Service Management.

The whole ISO/IEC 20000 family is under a major revision, due to better alignment with ITIL 3.0 and some ISO standards. Two additional parts are also under development. Part three presents the scope for certification in situations when the service is provided by multiple vendors with different kinds of subcontracting chains and agreements. Part four describes a process reference model (PRM). In addition, a new part to 15504 family, ISO/IEC 15504-8 defines a assessment model (PAM) which is based on ISO/IEC 20000-4.

ISO/IEC 20000-1 main content is divided in the general part (chapters 3 – 5) and 13 service delivery processes. The chapters 3-5 define requirements for service management system, process and service deployment and improvement, human resource management etc. The 13 processes in chapters 6 – 10 of the standard define the actual operational management and delivery of an IT service and are listed in Table 2.

Table 2: Processes in ISO/IEC20000-1. Note that chapters 3 – 5 are not included here because they are not recognised as processes in the standard.

Category	Process
Service Delivery processes	Capacity management
	Service level management
	Information security management
	Service continuity and availability management
	Service reporting
	Budgeting and accounting for IT services
Control processes	Configuration management
	Change management
Relationship processes	Business relationship
	Supplier management
Resolution processes	Incident management
	Problem management
Release processes	Release management

FiSMA has developed own process assessment model (called here FiSMA PAM) for assessing the capability of IT service management processes. The main use of FiSMA PAM is process improvement among member companies. FiSMA PAM is also an input to ISO work in defining ISO/IEC 15504-8. FiSMA PAM follows same architecture as the ISO/IEC 15504 Part 5 (known also as the original SPICE model). Process performance indicator types are base practices and work products. Capability levels 2 – 5 are the same as in 15504 Part 5. FiSMA PAM includes all the requirements from the ISO/IEC 20000-1 (when processes are assessed up to capability level 3) to allow assessment to be used as pre-certification check. It has also some additional practices based on the feedback from FISMA member community. ISO/IEC 20000-1 has two explicit processes outside of current CMMI-SVC scope, namely Budgeting and accounting for IT services and Information security management.

Chapters 3-5 of ISO/IEC 20000-1 are not yet expressed as processes in current version of FiSMA PAM. This work is going on in ISO standardization groups. Final versions of ISO/IEC 20000-4 and

15504-8 will have additional processes to cover chapters 3 – 5 of ISO/IEC 20000-1. The additional processes will cover for example service management, "Plan-Do-Check-Act" activities, service development, quality management and human resource management.

2.3 Mutual mappings and model comparisons

Mapping of CMMI-SVC and ISO/IEC 20000-1 (or FiSMA PAM) can be done at many levels of details. Here we present only one high level mapping from CMMI-SVC towards ISO/IEC 20000. The idea is to show what are the closest ISO/IEC 20000-1 elements of any single CMMI process. Result of this mapping is presented in Table 3. It includes 14 CMMI processes, because they all have some relevance in ISO/IEC 20000-1. Project management process areas PP and PMC are much more explicit in CMMI than in ISO20000.

We have done mappings also between more detailed model elements (like specific practices vs shall and should statements). Also mapping and comparison between exemplar work products in CMMI and documents and records in ISO/IEC 20000-1 is possible. Some model elements and concepts are quite different and they are listed in Table 4. Anyway, most of the processes and model elements can be mapped mutually quite easily and they have same meaning.

Table 3. Mapping of model elements, based mainly on CMMI purpose statements and specific goals vs "shall" requirements in ISO/IEC 20000-1. Mapping is done mainly from CMMI-SVC towards ISO/IEC 20000-1.

CMMI-SVC Process Area	**ISO/IEC 20000 element or process**
Organisational Service Management OSM	Chapter 3 Service Management System
Service System Development SSD	Chapter 5 Service Development
Project Planning PP	Chapter 4 "Plan"
Project Monitoring and Control PMC	Chapter 4 "Check", Service Reporting
Requirements Management REQM SG1	Business Relationships Management, Change Management
Requirements Management REQM SG2	Business Relationships Management, Service Level Management
Service Delivery SD	Chapter 4 "Do", Release Management
Service Transition ST	Release Management
Incident and Request Management IRM	Incident Management, Change Management
Problem Management PRM	Problem Management
Configuration Management CM	Configuration Management, Change Management
Capacity and Availability Management CAM	Capacity Management, Availability and Continuity Management
Service Continuity SCON	Availability and Continuity Management
Measurement and analysis MA	Service reporting (, Service Level Management)
Process and product quality assurance PPQA	Chapter 4 "Check"
-	Budgeting and Accounting for IT Services
-	Information Security Management

Both models have some fundamental concepts, and they are quite different in some areas. For example, CMMI-SVC has an idea of "Service System", which is basis for service delivery and most other processes. Service development is done by improving service system. Project management is important in CMMI-SVC, even service is rather continuous than discrete phenomena. Similar type concept in ISO/IEC 20000 is "Management System", including planning, monitoring and development of services.

ISO/IEC 20000 states which documents and records are required. In CMMI models the concept of work product is weaker. Instead of explicit requirements CMMI-SVC suggests typical work products. Both models have mostly same or same kind of work products. ISO/IEC20000 gives only the subject for the work product, whereas CMMI-SVC in certain processes has also a list of details that the work product might include.

Table 4 has a list of different concepts in CMMI-SVC versus ISO/IEC 20000-1. That list is based on our case study and feedback from participants. For example, we had long discussion about what "service system" really means and can it include for example human resources as one element. One fundamental concept in ISO/IEC 20000-1 is change management, whereas CMMI-SVC has included it in many processes.

Table 4: Some different concepts of CMMI and ISO20000. Note that a large number of quite similar concepts are not presented here.

CMMI-SVC key concepts	ISO/IEC 20000 key concepts
Organisational Service Management PA	Management system (including PDCA loop)
Service Requirement (Agreement)	Service level management and agreements (SLA)
Service System (SS)	Customer satisfaction
Standard Service	Change and Configuration management
SS Development and Transition	Documents and records
Project approach (project planning, project monitoring and control, risk management)	Continuous approach
	Well defined measurement objectives
	Post-implementation analysis

We did also more detailed mappings between models, for example between base versus specific practices. Another interesting area is work products. Full and detailed mapping between models is not possible to present in this article. Instead, we present one example process (Continuity Management) to express the idea of more detailed mapping. Table 5 lists some general terms for continuity management, and compares which corresponding elements each model has. The CMMI work product list is directly taken from the list of typical work products, while ISO/IEC 20000-1 column lists mandatory documents and records, plus work products not explicitly required.

Table 5: Processes and work products for managing the continuity of services.

Generalized concept	ISO/IEC20000-1	CMMI-SVC	Notes
Processes			
Continuity Management	Service continuity and availability management	Service continuity	ISO/IEC20000 covers availability in the same process, whereas CMMI has separate process for Capacity and availability management.
Work products, documents and records			
Service Continuity Plan (SCP)	Availability and service continuity plan	Service continuity plan	

SCP Training	--	SCP training material	In ISO/IEC20000 general requirements for training are covered in chapter 3.3.
Risk identification for SCP	Risk assessment	SCP typically includes "Identification of the threats and vulnerabilities that could impede the ability of the organization to deliver services"	CMMI has separate Risk Management process
Identifying and classifying the services for SCP	SLA	A business impact analysis (BIA) includes a description of the services provided.	
Identifying and classifying the information for SCP	Identified in Service description or in SLA	Legal and financial rights resources (service level agreements and contracts, organization legal operating charters, personnel benefit balances, payroll, and insurance records)	CMMI provides quite detailed list
Testing the SCP	Recorded test runs of continuity plans	Tests to exercise the SCP Description of environments necessary to execute the tests Results of the effectiveness of the service continuity tests Documented SCP test execution results analysis	
Improving SCP	Action plans based on continuity plan test failures	Suggested improvements to the SCP, collected from trainings Documented SCP improvement recommendations from tests Documented SCP test improvement recommendations	Improvement also in level 3 in CMMI and FiSMA PAM.
Maintaining the SCP **1. regularly** **2. in case of changing the existing or creating new services**	1. At least annual review 2. Done as part of change impact analysis	1. Updated SCP (e.g. annually or in major changes) 2. BIA describing the dependencies and interdependencies within the organizational infrastructure and any external assets and resources required for service continuity	2. An example of how the same idea can be expressed in a simple and in a complicated way
Resources and information needed in emergency situation	Contact lists Configuration management database	Key contact list Emergency operating resources (Orders of succession, Delegations of authority, Directory of critical personnel with contact information, Files and databases required to support identified essential service functions)	

3 Piloting units

The pilot units in TietoEnator were:

- A relatively small unit giving application management service for telecom operator, mainly in maintenance of intelligent network (IN) services. Their responsibility for customer was mainly "third-level support", for example correction of failures and defects in IN applications. They did also software development and testing services in IN domain, but that was left mainly outside of the SMMI-SVC and ISO/IEC 20000 pilot. Main instances of this pilot were fixed and mobile IN applications. The unit did not operate the production environment.
- A middle-size unit doing software development and application management service for banking and finance sector. This unit did earlier also production monitoring for the customer. After a major organizational change it focuses nowadays in consultancy service, application management and customer support.

IT service in both organizations is mainly consultancy and application management, including defect fixing and small improvements of IT applications. Production monitoring and analysis of operation failures was also partially in their service portfolio. Both units had also development services for the customer, but that was excluded from pilots (is covered better with CMMI-DEV model). Customer had selected a separate IT service company to operate and monitor IT services. The main responsibilities between suppliers and the customer is presented in figure 2.

Processes in both TE units were a mix of customer processes, own historical processes and TE corporate level processes. So, the CMMI and SPICE concepts of standard and defined process were not so clear. Both units had experiences also quite radical organisational changes just recently, and that made many organisational processes quite weak.

Also the scope of pilots was quite similar in both units. We identified 14 processes from CMMI-SVC to be relevant in pilots, and took all 13 processes from ISO/IEC20000. Table 3 lists all processes included in both pilots.

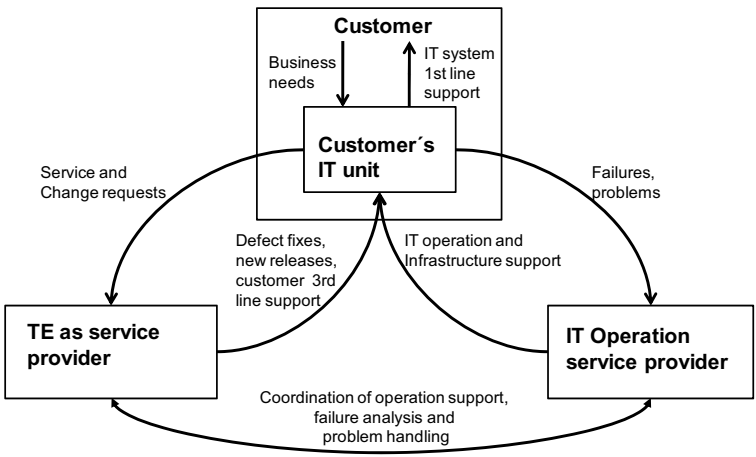


Figure 2. Main stakeholders of both pilots

Both pilots were done as parallel assessments, based on ISO/IEC 20000 and CMMI-SVC. The rigor of the assessment can be classified as "middle-size SPICE assessment" or "light Scampi B". The main focus was more in interviews and teamwork rather than detailed document review. We collected feedback from pilot units during the assessments.

Finally, we rated independently both CMMI-SVC and ISO/IEC 20000-1 processes. ISO/IEC 20000-1 rating was done by using FiSMA PAM model, which is fully conformant with ISO/IEC 15504-2 requirements. As a side results, we could compare also CMMI vs SPICE capability levels and different rating rules.

FiSMA has done also some additional features in both CMMI and SPICE models. Maybe the most important new concept is capability index CI, which is a sum of coverage measures of process indicators at each capability level. For example, if rating is done to capability level 3, the maximum value of CI can be 3. We have defined also rules, how to express CI results as "traffic lights" and comparisons with some reference or comparison values. Because these pilots were first in both units, we did not have any target level or value to compare with.

4 Results and findings from model comparison and piloting

Detailed comparison and analysis of CMMI-SVC and ISO/IEC 20000-1 assessment results is not possible to present in this article due to confidentiality. In general, both models worked reasonably well and gave useful results. In this article we present some findings from the usefulness of models.

FiSMA was interested about model similarities and differences, and how we could support our members in adopting them both with minimal overlap and additional effort. Because both models were still under development, we got also valuable and practical feedback to our standardization activities. Table 5 summarizes our model findings.

Table 5: The most important key characteristics and findings of the models raised in pilots.

CMMI-SVC key characteristics	ISO/IEC 20000-1 key characteristics
Processes are connected using "refer to xxx process area" links. there is no "central" process.	Processes are very tightly connected. Change management and configuration management tie all other processes together.
Practical approach, but some terminology is not everyday language. As in other CMMI constellations, no explicit documentation requirements.	Practical approach, for example terminology and documentation requirements were easy to understand both for practitioners and managers.
Basic assumption is project based operation. This needs interpretation in IT services.	Basic assumption is continuous service, maybe with periodic checks and updates.
Focus on service requirements and agreements, based on customer requirements.	Focus on business agreements (SLA), which are updated regularly with customer.
Processes for technical development and maintenance of services.	Processes for technical maintenance of service, but also for budgeting, security and customer relationship management.
Heavy and formal structure for process improvement. Same processes required as in CMMI-DEV, for example OPF and OPD.	Continuous process improvement. Easy to understand, but concept of capability levels is missing in the original ISO/IEC 20000-1.

The interpretation of model concepts was crucial in both pilots. For example, what service system really means in different type of service business logics? Some areas of models needed quite a lot additional clarification and interpretation in TietoEnator, according to these pilots:

- Capacity and availability, if main resources are people – how to interpret, predict and measure?

- Problem, problem management – at least when business responsibility is shared with customer and other stakeholders (what it means at different service lines (levels 1, 2, 3, 4) of customer support)
- Tailoring guidelines, are minimum requirements enough?
- Progress and milestone reviews, when service is continuous and reactive – what are necessary and required communication and meeting mechanisms
- If the production environment is managed by other stakeholder, continuity, availability and capacity management processes require close relationship to operator...

Usefulness of the models can be seen as main criteria for piloting units. As said many times, „all models are wrong but some of them are useful". Both TE pilots were interested only on process improvement. Based on that, we can define following criteria for usefulness:

1. Model should be easy to understand, both by practitioners and managers
2. Model should have „best practice" material, which can be used as source for improvement
3. Model should be suitable for and easily aligned with the business logic of each service
4. Model should be flexible, to include different amounts and combinations of processes for a certain business and contract type
5. Model should enable easy self-assessment to avoid unnecessary external costs and additional assessment effort

All these criteria got surprisingly similar results in both CMMI-SVC and ISO/IEC 20000-1 assessment and feedback. ISO model was easier to understand, because it was based more directly in ITIL. It was also more flexible than CMMI-SVC, allowing better adaptation in business logic of pilots. CMMI-SVC was a bit better in „best practice" thinking, because it contains also common processes of all CMMI constellations.

Business logic of both pilots was „time + material" based consultancy and application management. Especially CMMI-SVC was far too big model for that service type, and only some processes matched fully with the real life concepts and evidences. Also ISO model was too heavy, but mainly because its formal documentation and record requirements. See more detailed results in table 6. We used same rating scale NPLF here as in SPICE model. The result is based purely on judgement and feedback from units.

Table 6: Suitability of the model processes in TE pilot units.

TE-CMMI SVS Process Area			**ISO20000 requirement or process**
Project Planning PP	F		
Project monitoring and control PMC	F		
Organisational Service Management OSM	L		
Service system development SSD	L		
Service delivery SD	F		
Service transition ST	L	F	Release management
Requirements management REQM	L	F	Business relationships management
Incident and request management IRM	F	F	Incident management
Measurement and analysis MA	F	F	Service level management
Capacity and availability management CAM	P	P	Service continuity and availability management
Service Continuity SCON	P	P	Capacity management
Configuration management CM	F	F	Configuration management
Problem management PRM	P	P	Problem management

Process and product quality assurance PPQA	F	F	Chapter 3, 4, 5 “CHECK”
		F	Change management
		P	Budgeting and accounting for IT services
		F	Information security management
		F	Service reporting
		L	Chapter 3, 4, 5 “Plan”
		F	Chapter 3, 4, 5 “Do”
		L	Chapter 3, 4, 5 “Act”

Legend:		
	F	Full match and compliance
	L	Largely matching and compliant
	P	Partial match and compliance

5 Conclusions

The two pilots were not traditional service providers. They did not operate production environment nor help desks, but provided application management and consulting services. In that sense it was interesting to see how the companies adapted the ideas from the models.

As stated earlier, feedback from the assessment was asked already during the interviews and in reporting sessions, as well as afterwards. Generally the feedback was good, but because of the nature of the units assessed, especially the CMMI model was thought to be quite heavy and abstract. Most of the concepts in the models were easily understood.

Both models are changing. So, the value of these pilots is only temporary. More case studies and pilots are needed in future, when models evolve. Our pilots were useful also as inputs to standardisation work. It is obvious that similar types of model validation are needed also in future.

6 References

1. ISO/IEC 20000:2005 Information technology — Service management — Part 1: Specification
2. CMMI for Services, draft version 0.5
3. ISO/IEC15504-5:2006 Information Technology — Process Assessment — Part 5: An exemplar Process Assessment Model
4. ISO/IEC15504-2:2003 Information Technology - Process Assessment – Part 2: Performing An Assessment.

7 Author CVs

Mika Johansson

Executive Director in FiSMA, partner and consultant in Spinet Oy.

Mr Mika Johansson has been working as a consultant in quality and project management areas for more than ten years. He has been Executive Director in Finnish Software Measurement Association FiSMA since 2006. In Spinet Oy, Mika has performed assessments and audits using SPICE, CMMI, ISO9001, ISO/IEC 20000 and nuclear specific models

since 2003. He has actively involved in ISO/IEC based assessment model development in FiSMA and ISO, and is certified ISO/IEC 20000 auditor.

Risto Nevalainen

Senior Advisor in FiSMA, part-time researcher in Tampere University of Technology Pori Unit.

Mr. Risto Nevalainen (Lic. Tech.) has long experience in software measurement and quality topics. He has been managing director of STTF Oy for last 10 years. His working experience includes position as managing director of Finnish Information Technology Development Center during 1989-1995. Before that he had different research and management positions for example in Technical Research Centre (VTT), Technical University of Helsinki (HUT), Finnish Prime Minister's Office and Finnish Economic Planning Centre. Mr. Nevalainen has participated in ISO15504 (SPICE) standard development since beginning. He is Competent SPICE Assessor and ISO9000 Lead Assessor.

Process Similarity Study: Case Study on Project Planning Practices Based on CMMI-DEV v1.2

Jose A. Calvo-Manzano, Gonzalo Cuevas, Mirna Muñoz, Tomás San Feliu
Faculty of Computer Science, Polytechnic University of Madrid Campus de Montegancedo, Boadilla del Monte, 28660 Madrid, Spain
{jacalvo, gcuevas}@fi.upm.es, mamunoz@mpsei.fi.upm.es,tsanfe@fi.upm.es

Abstract

This paper shows a method for identifying the similarity among standards and models of best practices. The standards and models considered were CMMI-DEV v1.2, PMBOK, TSP, PRINCE2, COBIT, ISO 9001:2000, and ISO/IEC 15504, because they are the most widespread in the world. The method is focused on the practices of project planning process.

This study is the first phase, consisting of a methodology development, with the goal of incorporating process improvement by using basic components that enable a smooth and continuous improvement of the organization's processes capacity level. This would avoid initial resistance to change and the subsequent problems. A case study is presented using the project planning process area from CMMI-DEV v1.2 as reference.

Keywords

Project planning, best practices, CMMI-DEV, PMBOK, TSP, PRINCE2, COBIT, ISO/IEC 15504, ISO9001:2000

1 Introduction

Royce Walter mentions that “project management is the art of balancing competent objectives, risk management and overcome restrictions on delivering a product where both client and user are perfectly aware of the needs …" [1]. In this context, project management will ensure the success of the project development.

However, recent reports such as the “Chaos Report 2006” [2] by the Standish Group, where improvements in performance experienced by organizations through recent years were analyzed, revealed that projects percentage categorized as “successful” (completed on time, on budget and known user requirements) was only 35%, even though there was an increase compared with data showed in the “Chaos Report 1994” [2] where the successful percentage was 16.2%.

The previous paragraph makes clear the necessity of improving project performance and the importance of project planning. In this context, the purpose of planning is to establish and maintain plans that define project activities [3]. As a result, the following questions arise: 1) which model do we follow for project planning improvement? and 2) have current standards and models similarities?

The reason for finding similarities in processes practices in the most widespread models and standards in the public and private sector is to establish the first phase of a research work. The research work consists of the development of a methodology whose goal is to incorporate elemental process improvement components. The methodology would enable a smooth and continuous improvement of the capacity level of the organization’s processes depending on their business goals. This would avoid initial resistance to change and the subsequent problems.

Changes that are too great or too rapid can overwhelm the organization, destroying its investment in learning represented by organizational process assets. Too many changes at the same time are a barrier to change. Besides, there must be compatibility on the improvement with the existing processes values and skills of potential end users.

This paper is divided into four sections. Section two covers the similarity method description; section three describes the project planning practices case study and section four presents the conclusions

2 Similarity Method

At the present, there are several results of similarities available: Mapping TSP to CMMI, ISO 9001:200 y CMMI v1.1 [4], Mapping OGC PRINCE2 to SEI CMMI 1.1 [5], CMMI and PMBOK Mappings [6]. All of them provide only the mappings results and not the method followed to get the results. Besides, they are only bilateral mappings. The Models and Standards Similarity Study method (MSSS) shows how to get the similarities, in addition to providing a multiple comparative.

MSSS method was developed after an exhaustive analysis of available mappings studies previously mentioned was done. It provides minimum steps for finding the similarities among the models and standards considered within the scope on the study.

2.1 Select the Models and Standards to Be Analyzed

The models and standards were selected focusing on: 1) models which contain the target processes, 2) models which have a wide use in the target processes and 3) models with public information.

2.2 Choose the Reference Model

Before studying models and standards it is essential to know where to address the research effort. Therefore, it is necessary to choose the model that best serves as reference to address this effort, which is obtained by the mapping among models and standards and choosing the model that has wider coverage in relation to target.

2.3 Select the Process

A process is a cluster of related practices that, when implemented collectively, satisfy a set of goals considered important for improvements in the process selected [3].

If several processes are considered to satisfy the target, then the priorities will be established according to dependencies among them.

The process selection will be useful to establish the study scope.

2.4 Establish the Detail Level

Studying models and standards implies handling large amounts of information. Usually the structure of information used by models is different. Therefore, the difficulty of determining the appropriate level of detail should be kept in mind.

A high level analysis may not provide enough insight into similarities and differences, and a low level analysis could result in an overwhelming number of interrelationships which also fails to properly identify the correspondence among models [4].

MSSS proposes to analyze the information contained in the models and standards and then to make a glossary. The glossary will establish how one element is referenced in each model.

Consequently, it will be possible to analyze the structure of the models and standards selected and to decide the detail level that: 1) all models and standards contains and 2) contains enough information to cover the expectations of the study.

2.5 Create a Correspondence Template

In order to establish an appropriate comparison among models and standards, a template was created. The template should capture the correspondence information among models and standards.

The template is generated by taking first the information provided by the reference model. Second, the information provided by other models and standards is added.

2.6 Identify the Similarity among Models

The similarity is the correspondence among the information provided about how to carry out an activity with different models. The similarity allows us to establish what information in a model can strengthen the other models. In MSSS three questions were formulated to identify the similarity:

- *Is there any information, "item a", in the X model related to the reference model?*
- *What is the information?*
- *What is the additional information provided by X model that could help to carry out "item a" in the reference model?*

Where:

- *X is each of the models and standards analyzed, and*
- *a is the element analyzed*

After answering these questions, the template was completed with the information obtained from the models. First of all, the information proposed by the reference model was introduced in the template and then the proposed information by the other models.

Once the template was completed, the information of the template was analyzed, verified and refined, in order to confirm that it really contains all information on how to do any activity or process to achieve an objective.

2.7 Show Obtained Results

An important factor to understand the obtained results is the way they are shown. During this research it was identified that the best practice to show this kind of results is by using a table because the information is stored in a structured way.

Therefore, the established template in step 2.5 is structured as a table and its items are ordered so that it helps to understand the correspondence of information among models.

3 Implementing the MSSS Method in Project Planning

This section shows the implementation of the MSSS method in project planning at *everis Consultants*.

3.1 Select the Models and Standards to Be Analyzed

The models and standards analyzed were chosen based on a survey carried out by TeraQuest Metrics, Inc [7]. The survey was done to determine which models and standards were available for assuring that projects are successfully completed on time, within budget and with the expected benefits [7].

Besides, the analyzed models and standards were proposed by 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). They all provide best practices reference frameworks [7].

According to Mc Grath [8], Turner [8], and Lawes [9], a best practice could be a management practice, a technique practice, a tool, or a methodology, which has proved a right performance and improvement in one or more aspects widely accepted by industry such as productivity, cost, schedule, or customer satisfaction. Then, talking about best practices is “learning from others”.

The analyzed standards and models are:

- *Capability Maturity Model and Integration for Development 1.2 (CMMI-DEV v1.2)*: it is a reference maturity model of process improvement that helps organizations in products and services development. It covers the development and maintenance activities applied to both products and services through the product life cycle [3], [10].
- *Control Objectives for Information and related Technology (COBIT)*: it is an open standard, whose structure provides good practices across a domain and process framework, and presents activities in a manageable and logical structure [11].
- *ISO 9001:2000- Quality Management System*: it is an international standard which is focused on quality management system effectiveness to carry out customers’ requirements. It covers quality

system requirements that support all product lifecycle including deliverables, initial agreements, design, development and support of the product [4], [12], [13].

- *ISO/IEC 15504 Information technology – Process assessment*: it is an international standard that provides a structured approach for the assessment of processes. In part 5 of this standard "An assessment model and indicator guides", contains a guidance of good software engineering base practices for each process such as customer-supplier, engineering, support, management and organization [14].
- *Project Management Body of knowledge (PMBOK)*: it is a reference model which covers project management processes, tools and techniques by providing a set of high level business processes to all industries [15].
- *Projects IN Controlled Environments (PRINCE2)*: it is a structured methodology based on the experience of project management scores. It provides a structured method for effective project management in all types of projects and a common language for all participants in the project [5], [16].
- *Team Software Process (TSP)*: it is a defined process which guides development teams in producing high-quality software-intensitive systems [17]. It provides a detailed guidance and scripts for guiding development teams and their management in planning and developing quality products on predictable schedules [18].

3.2 Choose the Reference Model

CMMI-DEV v1.2 model was chosen as the reference model because it provides a way to manage an integrated approach to development activities as part of achieving their business objectives [3].

CMMI-DEV v1.2 provides best practices that address development and maintenance activities applied to products and services covering the product's lifecycle practices from conception through delivery and maintenance. The structure items are: process areas, specific and generic goals, and specific and generic practices. Fig. 1 shows CMMI-DEV v1.2 structure. The structure items are: process areas, specific and generic goals, and specific and generic practices.

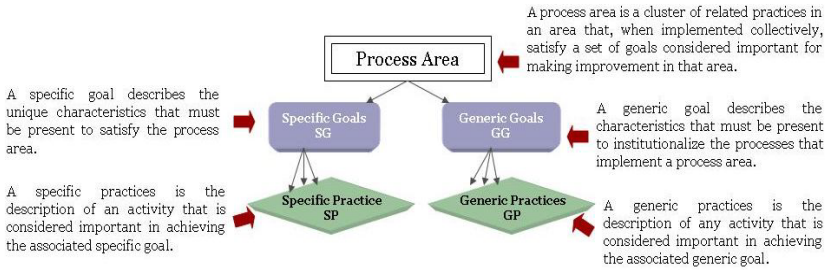


Fig. 1. Structure of CMMI-DEV v1.2 components

3.3 Select the Process

The Project Planning process was chosen because it is considered critical and key to project management success [16]. As mentioned in the introduction, recent reports show that organizations still faces serious problems on project planning.

The CMMI-DEV v1.2 Project Planning process area is a simulation of how things should happen in future, which makes it an essential component for projects [19]. The purpose of project planning, according to CMMI-DEV v1.2, is to establish and maintain the plans that define the project activities [3]. Table 1 shows the three goals that are covered in this process area.

Table 1. Project planning process area specific goals and practices.

Specifi Goal	Spcific practices
SG1 Establish Estimates: Estimates of planning parameters (size, effort and cost) are established and maintained.	*SP 1.1 Estimate the scope of the project.* *SP 1.2 Establish estimates of work products and task attributes.* *SP 1.3 Define project lifecycle.* *SP 1.4 Determine estimates of effort and cost*
SG2 Develop a Project Plan: A project plan is established and maintained as the basis for managing the project. It includes items such as budget, schedule, resources, needed knowledge and skills, risk, data management and stakeholders' involvement	*SP 2.1 Establish the budget and schedule.* *SP 2.2 Identify project risks.* *SP 2.3 Plan for data management.* *SP 2.4 Plan for project resources.* *SP 2.5 Plan for needed knowledge and skills.* *SP 2.6 Plan stakeholder involvement.* *SP 2.7 Establish the project plan*
SG3. Obtain Commitment to the Plan: Commitments to the project plan are established and maintained. It includes review plans that affect the project, reconcile work and resource levels and obtain commitment	*SP 3.1 Review plans that affect the project.* *SP 3.2 Reconcile work and source levels.* *SP 3.3 Obtain plan commitment.*

3.4 Establish the Detail Level

The detail level is an important factor in highlighting the identified correspondences or additions among the analyzed standards and models with respect to the reference model.

After the information was analyzed and the glossary was prepared, it was decided to establish the level of mapping at specific practice level because:

1. A specific practice level can be found in all analyzed models and standards, and

2. The specific practices help organizations in their process improvement.

Fig. 2 shows the correspondence of the specific practice concept among the analyzed models and standards.

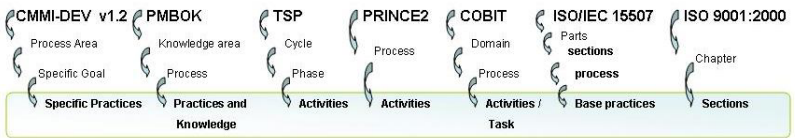


Fig. 2. Specific practices correspondence among analyzed models and standards

3.5 Create a Correspondence Template

As we mentioned in 3.1, a best practice could be a management practice, a technique practice, a tool, so the template should allow the organization to capture any of these elements. A template was designed based on the CMMI-DEV v1.2 structure. The CMMI-DEV v1.2 structure includes: work products, subpractices, informative components [3]. After, other items were added from the other models and standards such as inputs, tools and techniques. The final template covers.

- ***Inputs***: All input information related to the specific practice is included.
- ***Subpractices***: Subpractices proposed by the models for each specific practice are included. The subpractices are labelled by adding the specific practice number (e.g. SP1.1, SP1.2…SPx.y) and a consecutive number (1, 2…n).
- ***Tools and techniques***: All information related to tools and techniques are included. The tools and techniques are labelled by adding an H if it is a tool or T if it is a technique, followed by the specific

practice number and finally a consecutive number that identifies the sequence when the specific practices has more than one tool or technique (e.g. H1.1.1, T.1.2.4…n).

- ***Work products***: All work products information obtained by performing a specific practice is included.
- ***Informative Components***: All information which helps to perform a specific practice with success is included.

3.6 Identify the Similarity among Models

The similarities study was done as indicated in step 2.6 of MSSS method. The questions were answered by doing an exhaustive review of each practice of the different models and standards. Table 2 shows an example of the questions designed for the specific practice "SP2.2 Identify Project Risks" and the PMBOK model as well as a summary of their answers.

Table 2. Example of MSSS questions and their answers on *"SP2.2 Identify Project Risks"* and the PMBOK model.

Questions	Answers
Is there any information in the PMBOK model which identifies project risk?	Yes
What information is it?	*Risk Identification*: Involves determining which risk might affect the project and documenting their characteristics. *Qualitative Risk Analysis*: Involves performing a qualitative analysis of risk and conditions to prioritize their effects on project objectives.
What is the information that helps CMMI to carry out the identification of projects risk?	*Input*:: Risk management plan, project planning outputs, Risk categories, Historical information, identified risk, project status, project type, data precision, scales of probability and impact, assumptions. *Tools and techniques*: documentation reviews, information-gathering techniques, checklist, assumptions analysis, diagramming techniques, risk probability and impact, probability/impact risk rating matrix, project assumptions testing, data precision raking.

3.7 Show Obtained Results

A template example of similarities among CMMI-DEV v1.2 and other analyzed models were selected. The template contain basic information on how CMMI-DEV v1.2 could be strengthened by the other analyzed models and standards, which result in an input of best practices to be used in the incorporation of process improvement elemental components methodology. Table 3 shows the results for "SP2.2 Identify project risk".

The templates are not an implementation guide but they could be used by the methodology as a guide to best practices and correspondence indicators among models and standards.

4 Conclusions

The goal of this study was to identify the similarities among models and standards of project planning practices in order to establish the first phase that consists of the development of the methodology whose goal is to incorporate elemental process improvement components. The methodology would enable a smooth and continuous improvement of the capacity level of the organization's processes depending on their business goals.

Table 2. Template of *"SP2.2 Identify Project Risks"*

PP SP 2.2 Identify Project risk			
Inputs	*Subpractices*	*Tools and techniques*	*Work products*
Historical risk management plan (PMBOK) (ISO/IEC 15504) Project planning outputs (PMBOK) (PRINCE2) (ISO/IEC 15504) Project charter Work Breakdown Structure (WBS) Project description Schedule and cost estimates Resource plan Procurement plan Assumption and constraint list Risk categories(project management, organizational and external risk) (PMBOK) Historical information (PMBOK)	SP2.2.1 Identify risks (CMMI) (ISO/IEC 15504)	T2.2.1 Structured interviews (CMMI) T2.2.2 Brainstorming (CMMI) T2.2.3 Documents review (PMBOK) T2.2.4 Delphi (PMBOK) T2.2.5 Strengths, weaknesses, opportunities, and threats (SOWT) analysis (PMBOK) T2.2.6 Risk taxonomies (CMMI) T2.2.7 Risk assessments (CMMI) (TSP) T.2.2.8 Checklist (Checklists) (CMMI) T2.2.9 Performance models (CMMI) T2.2.10 Cost models (CMMI) T2.2.11 Network analysis (CMMI) T2.2.12 Quality factor analysis (CMMI) T2.2.13 Diagramming techniques (PMBOK) H2.2.1 Fishbone Diagram (PMBOK) T2.2.14 Risk probability and impact (PMBOK) T2.2.15 Probability / impact risk rating matrix (PMBOK)	Identified risks (CMMI) (TSP) (PRINCE2) Risk impacts and probability of occurrence (CMMI) (ISO/IEC 15504) Risk priorities (CMMI) (PMBOK) Triggers (PMBOK) Overall risk ranking for the project (PMBOK) Trends in qualitative risk analysis result (PMBOK) ITL (Issue Tracking Log) form (TSP)
Identified Risk (PMBOK) Project status (PMBOK) Project type (PMBOK) Probability / impact risk rating matrix (PMBOK) Data precision raking (PMBOK) Assumptions (PMBOK)	SP2.2.2 Document the risks (CMMI) SP2.2.3 Review and obtain agreement with relevant stakeholders on the completeness and correctness of the documented risks (CMMI) SP2.2.4 Revise the risks as appropriate (CMMI)	T2.2.16 Project assumption testing (PMBOK) T2.2.17 Data precision raking (PMBOK) T2.2.6 Risk taxonomies (CMMI) T2.2.7 Risk assessments (CMMI) T2.2.9 Performance models (CMMI) T2.2.10 Cost models (CMMI) T2.2.11 Network analysis (CMMI) T2.2.12 Quality factor analysis (CMMI) T2.2.13 Define risk management checkpoints and responsibilities (TSP) T2.2.14 Risk probability and impact (PMBOK)	

Informative Components

- Participants in risk identification generally include as possible: project team, risk management identification, subject matter experts from others parts of the company, customers, end users, other Project managers, stakeholders, and outside experts (PMBOK)
- The organization must plan and develop the needed procedures for product realization. The procedures must be consistent with the quality management plan (ISO 9001:2000)
- Management must plan and monitor product design and development, because responsibilities and empowerment should be determined (ISO 9001:2000)
- The risk identification process should include qualitative and, where possible, quantitative risk ranking and should obtain input from management brainstorming (COBIT)
- The risk assessment approach should focus on the examination of the essential elements of risk and the cause/effect relationship between them. The essential elements of risk include tangible and intangible assets, asset value, threats, vulnerabilities, safeguards, consequences and likelihood of threat (COBIT)
- The risk assessment should consider business, regulatory, legal, technology, trading partner and human resources risks (COBIT)
- Allocate to each high risk or critical activity a resource in which management has confidence (PRINCE2)
- Monitor the schedule and quality of any external product to be delivered on which any activities in the plan are dependent (PRINCE2)
- The addition of risk management activities will elongate the schedule and require extra resources. The benefit of the protection against risk is valuable, but remember to allow for the cost of these activities in the plan (PRINCE2)
- Identify risk to the project both initially within the project strategy and as they develop during the conduct of the project (ISO/IEC 15504)
- Assess the probability of occurrence, impact, time-frame, causes and interrelationships of risk for determining the priority in which to apply resources to mitigate these risk (ISO/IEC 15504)
- Roles involved: team leader (TSP)

As a result of MSSS method application to project planning practices:

1. The analyzed models and standards are: CMMI-DEV v1.2, PMBOK, TSP, PRINCE2, COBIT, ISO 9001:2000, and ISO/IEC 15504,

2. The target is project planning as it is considered a critical area in project management,

3. The detail level of the study was at specific practice levels because a specific practice level can be found in all analyzed models and standards,

4. A template was defined to represent the results. The template items are: input, supractices, tools and techniques, work products and informative components.

With the implementation of the method, it was initially concluded that, the analyzed models and standards had a similar scheme for the Project Planning process. However, each one provides in some aspects more detailed information.

Second, a guide to best practices to be used as input information in the methodology research previously mentioned was obtained. This guide has been validated and successfully tested in *everis Consultants*. The guide enables the selection of external best practices that when used by the methodol-

ogy, would enable a smooth and continuous improvement of the capacity level of the organization's processes depending on their business goals. This would avoid resistance and subsequent problems.

The guidance also, provides a proposal on how to strengthen CMMI-DEV v1.2 using information from other models and standards was made.

Fig. 6 shows a proposal on how CMMI-DEV v1.2 model can be strengthened with information from other analyzed models and standards.

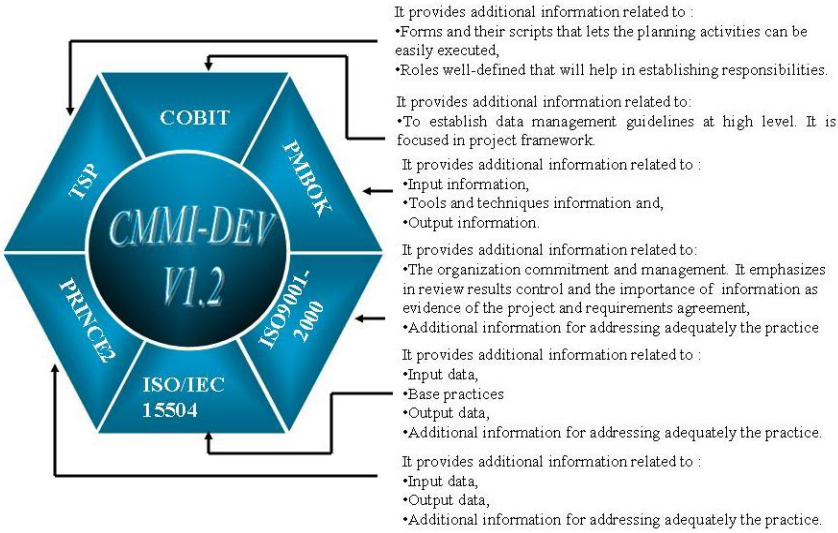


Fig. 6. Proposal of the relationship among models and standards

Finally, the main issues that arise when an organization decides to implement one model or standard are:

1) The great amount of models and standards available in the market.

2) The high-cost of assimilating the appropriate knowledge by the organization and its deployment.

Therefore, the MSSS method would help organizations to choose the best practices compliant with the available models and standards in the market.

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 Latin American Region.

5 Literature

1. Royce, W.: Software Project management: a unified framework. (eds.) Addison-Wesley Longman, Inc. (1998)
2. Rubinstein, D.: Standish Group Report: There's Less Development Chaos Today. In: SD Times Software Development. http://www.sdtimes.com/article/story-20070301-01.html (2007)
3. Chrissis M., Konrad, M. & Shrum, S.: CMMI Second Edition: Guidelines for Process Integration and Product Improvement. (eds.) Addison Wesley, pp.3-4, 31-42, 401-425 (2007)
4. Mustafelija, B. & Stromberg, H.: ISO 9001:2000 - CMMI v1.1 Mappings. In: Software Engineering Institute (2006)
5. Luqman, A., Hussain, F. & Tauseef-ur-Rehman, S.: Mapping OGC PRINCE 2 to SEI CMMI 1.1. In: Information and Communication Technologies: 2005 (ICICT 2005). ISBN: 0-7803-9421-6 (2005)
6. Sherer, W., & Thrasher, S., CMMI and PMBOK Mappings, https://seir.sei.cmu.edu/seir/domains/FRMSET.DOCUMENTS.NAV.PDF.ASP?ACTION=view&DOMAIN=CMMi&SECTION=General&SUBSECTION=Mappings&SUBSUBSECTION=NONE&DOC=BSCW-PMBoK-and-CMMI-Mappings&MEDIA=Paper&IMG=&MAX=0&LIMIT=All
7. Brotbeck, G., Miller T. & Statz, J.: A survey of Current Best Practices and Utilization of Standards In the Public and Private Sectors. In: DIR: Department of Information Resources (2006)
8. Walter, E: Implementing Best Practices in the Joint Battlespace infosphere (JBI) Program at AFRL. In: 2003 Presentations-Conference on the Acquisition of Software-Intensive Systems (2003)
9. ITSMF ARGENTINA: the best practices, http://itsmf-argentina.com.ar/mejores.php3
10. Persse, J.: Project Management Success with CMMI. (eds.) Prentice Hall, pp.57-95 (2007)
11. IT Governance Institute: COBIT Mapping Overview of International IT Guidance 2nd Edition. In: ISACA Serving IT Governance Professionals (2006)
12. Mustafelija, B. & Stromberg, H.: Exploring CMMI-ISO 9001:2000 Synergy when Developing a Process Improvement Strategy. In: SEPG 2003 (2003)
13. de la Villa, M., Ruiz, M., Ramos, I.: Modelos de Evaluación y Mejora de Procesos: Análisis Comparativo. In: 5th Adis 2004 Workshop on Decision Support in Software Engineering (2004)
14. AENOR. ISO/IEC 15504: 2004 Information technology – Process assessment. first edition 2004 (2004)
15. IEEE Computer Society: IEEE Guide Adoption of PMI Standard A Guide to the Project Management Body of Knowledge: IEEE Guide Adoption of PMI Standard A Guide to the Project Management Body of Knowledge. IEEE Std 1490-2003 (Revision of IEEE Std 1490-1998). ISBN: 0-7381-3894-0 (2004)
16. Office of Government Commerce (OGC): Management Successful Projects with PRINCE2. (eds.) TSO 2005, pp. 137-150, 167-190 (2005)
17. McHale, J. & Wall, D.: Mapping TSP to CMMI. Technical Report: CMU/SEI-2004-TR-014 (2005).
18. Humphrey, W.: TSP Leading a Development Team. (eds.) Addison-Wesley, 2005, pp. 71-91, 307 (2005)
19. Cuevas, G., de Amescua, A., San Feliu, T., Arcilla, M., Cerrada, J. A., Calvo-Manzano J. A. & García, M.: Gestión del Proceso Software. (eds.) Universitaria Ramón Areces, 2002, pp. 472 (2002)

6 Author CVs

José Antonio Calvo-Manzano Villalón

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.

Gonzalo Cuevas Agustín

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.

Mirna Ariadna Muñoz Mata

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 elemental process improvement components. She is colaboring with an everis consultants research project.

Tomás San Feliu Gilabert

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.

Strengthening Maturity Levels by a Legal Assurance process

Luigi Buglione, Ricardo J. Rejas-Muslera, Juan José Cuadrado Gallego

Abstract

One of the key elements for the viability of information system projects is given by the adoption of legal assurance activities and measures since nowadays they can arise legal risks that, in some cases, can suppose a serious threat for project commercial and financial success. When calculating the Return of Investment (ROI) for a software process improvement initiative, readers would not take care which are the cost issues impacting on such values, supposing the activities generating such value are referable only to the processes included in a maturity model (MM) such as CMMI or ISO 15504. During last years, moving from the initial Philip Crosby's idea for measuring and checking the organizational evolution of an organization, a plenty of MM have been created, but there is no news about a legal assurance process that make more systematic the way legal risks are (or should be) managed. On the other hand, professional practice usually does not incorporate standardized processes in order to discipline the legal assurance activities and measures, returning a feeling for a lack of project legal security.

This paper proposes to take care of a Legal ASsurance process (LAS) as an additional process area within a MM, in order to provide a suitable instrument for the management of inherent legal risks to any information systems project. After presenting main elements for this new process, it will be presented using the typical CMMI process area architecture, where it would be configurable as a Support process at Maturity Level 2 (ML2).

Keywords

Legal Audit, Maturity Models, CMMI, ISO 15504, Maturity Level 2, Support Process.

1 Introduction

The ever increasing relevance of software systems in all economic and social sectors implies an increment in the importance of legal aspects associated with such systems. Legal aspects are not only related with the end product, but they are also related with each activity performed through the software development lifecycle and its related intermediate products, from the early phases till the project closure [1]. Even if there are no official statistics on the cost of litigation in ICT organizations, there is a diffused perception – verifiable on an emerging body of technical literature – about the need to manage (and not only run) an organic legal process within the software project lifecycle. A new emerging context that would greatly benefit from such activities is the Open Source Software (OSS) world, recognized also by Gartner Group as one of the five hottest IT topics and trends in 2005 for next five years [2]. Among the plenty of possible licenses [3], a particular attention will be paid more and more in the near future to Lesser General Public License (LGPL), a license type between the strong-copyleft GNU General Public License (GPL) and the permissive licenses such as the BSD licenses or the MIT License. Written in 1991 and updated in 1999, LGPL places copyleft restrictions on the program itself but does not apply these restrictions to other software that merely links with the program. There are, however, certain other restrictions on this software. Its primary use is intended for software libraries, although it is also used by some stand-alone applications (e.g. Mozilla and OpenOffice). From a management viewpoint, the consequence is a growing attention to be paid to legal issues for enforcing intellectual property [4-9], even if the huge work done by WIPO during years [10]. Anyway, it must be observed that OSS is not the solely issue/environment where legal costs have a huge impact. Of course, being it a quite young field, there are new, unfaced issues to solve, but it is a general issue, as also evincible from [8], analyzing the Total Cost of Ownership (TCO) for open/proprietary systems, where the item “Legal costs” is a diffused activity to be performed.
Again, the need to be compliant with Sarbanes-Oxley (SOX) Section 404 requirements [11] added a further request for legal skills within ICT organizations and a lot of publications and researches have been published during last years about the compliance with most known maturity models as the Capability Maturity Model Integration (CMMI) [12], where 4 out of 6 of SOX control objectives are addressed by ML2 process areas [13][1].
Thus, according to a more complex context in the ICT world, legal aspects should be considered as a new and more and more relevant kind of project risk that without adequate management can increase the possibility of failure of a project.
The objective of this paper is to analyze the impact and return on investment (ROI) of a Legal Assurance process included within ICT organization and present a proposal for a Legal Assurance (LAS) process, which defines legal audit activities that must be performed as part of a software process assessment and improvement model with the objective of minimizing legal risks for software projects. In the following sections, the proposal is formulated according to the CMMI architecture [14] and terminology [15], but it can be easily represented according to any other process model framework. The aim is to provide industry with a framework to manage legal risked inherent to all software projects efficiently. Such a framework allows an ICT organization to move from a reactive risk strategy to a proactive one.

The paper is structured as follows: Section 2 shows related works on this issue, stressing the relevance of a legal assurance process. Section 3 proposes the high-level view for a Legal Assurance (LAS) process in a CMMI-like style, listing the goals and practices for such Process Area (PA). Finally, Section 4 draws main conclusions and proposes outlines for future works.

2 Related Works

Searching for related works in the technical literature on the wave of the so-called “maturity models mania” [16], it does not exist a specific and organic formulation neither for a “Legal Risk Maturity

[1] The four control categories covered are: Change Control process, Emergency Changes, Project Life Cycle, Testing, while “Application Logical Access Control” .and “Access Administration Control” are not covered by CMMI.

Model" nor for a process to be considered within the scope of one yet existing process model. Logical links in terms of content would be to "risk", "risk management" and "contract management" [1].
Looking at Risk Maturity Models, INCOSE's Risk Management Maturity Model (RMMM) [17], composed as the original Crosby's Quality Management Maturity Grid [18] with a grid crossing four maturity levels (1: ad-hoc; 2-initial; 3-repeteable; 4-managed) and five dimensions (definition; culture; process; experience; application), there is not an explicit mention of legal risks. The same happens for RIMS RMM for Enterprise Risk Management [19] and the IACMM CMM [20] on contract management.
Looking at project management MM such as Project Management Maturity Model (PMMM) [26], Portfolio-Programme-Project Maturity Model (P3M3) [21], Prince 2 Maturity Model (P2MM) [22] and PMI Organizational Project Management Maturity Model (OPM3) [23] also here there is not specific room for legal issues, but it could be included within the Risk Management process.
Nevertheless, the most important software process assessment and improvement models such as the Capability Maturity Model Integration (CMMI) [15] or the correspondent ISO standards on process models (12207 for Software Engineering [24] and 15288 for Systems Engineering [25]) and the related assessment model provided by the 15504 series [26], do not properly include specific processes for legal audits. Both SEI and ISO process models were recently extended, including a new group of processes, called "Acquisition":

- in the ISO/IEC 15504 process exemplar model is composed by five processes (ACQ.x)2, introduced with the two amendments dated 2002 and 2004;
- in CMMI, a new constellation (CMMI-ACQ [27]) was released on November 2007, including five processes placed between ML2 and ML33.

and in both cases such goals practices are seen from an acquirer's viewpoint, while the object of this proposal is the way to manage legal issues within an organization developing software & systems.

Looking at CMMI-DEV v1.2, it is possible to find scattered mentions to contractual or legal aspects in:

- Supplier Agreement Management (SAM, SP1.3 – Establish Supplier Requirements);
- Requirement Development (RD, SG1 – Develop Customer Requirements) process areas;
- Risk Management (RSKM, SP1.1 – Determine Risk Sources and Categories).

but it is not specified when and how activities for legal risks management should be carried along the phases of the software development lifecycle. Further evidence can be verified looking at the CMMI glossary, under the item "Process Architecture", where *contract management* is the example explicitly cited of an external process against the current formulation of the CMMI process model.

This implies that activities performed in such area depend on the managers' perception of risks. Generally speaking, such activities do not follow any temporal pattern to systematically perform them and the most common activity only consists of performing a Due Diligence or legal audit before marketing the product. This lack of a standardized process means that legal risks are handled reactively instead of proactively, which implies that the problem has already happened and little or nothing can be done about it.

3 A proposal for a legal assurance process

In order to properly manage all legal implications of a software project throughout its development lifecycle, firstly there is the exigency to define a legal audit software process with all needed activities. On one side, such a process must take into account legal risks that may affect the project, and on the other side, it must specify what actions can be adopted to avoid or minimize such risks and when.

[2] ACQ.1: Acquisition Preparation; ACQ.2: Supplier Selection; ACQ.3: Contract Agreement; ACQ.4 : Supplier Monitoring ; ACQ.5 : Customer Acceptance.
[3] ML2: Agreement Management (AM); Acquisition Requirements Development (ARD). ML3: Acquisition Technical Management (ATM); Acquisition Validation (AVAL); Acquisition Verification (AVER).

In this paper, the CMMI-DEV process model [15]4 and architecture [14] was used as a starting point to describe this proposal.

CMMI glossary defines a process as "*the action of defining a process*", and specifies that the process description is "*a documented expression of a set of activities performed to achieve a given purpose that provides an operational definition of the major components of a process. The documentation specifies, in a complete, precise, and verifiable manner, the requirements, design, behaviour, or other characteristics of a process. It also may include procedures for determining whether these provisions have been satisfied. Process descriptions may be found at the activity, project, or organizational level*".

Moving from these definitions, our proposal for a legal assurance process consists of the description of its process:

1. **Purpose definition**. This process has 2 main objectives:

   a. Optimization of business opportunities;

   b. Management of the legal risk along the SLC phases.

2. Document **a set of activities** and specify the requirements, the design, and the behavior and other characteristics of a process in comprehensive, precise and verifiable.

3. It is also possible to include **a quantitative process** to verify the extent of the activities set in place.

The legal assurance (LAS) process has been fully formulated and described in [28][29] with a generic format. The objectives in this paper are:

- to produce its high-level reformulation in a CMMI-like style for the process part;
- to properly allocate LAS to a maturity level (ML) and possibly to a representation type, arguing the reasons for those assumptions.

3.1 Elements to be mapped in the CMMI language

In [28][29] the proposal for a legal assurance process was formulated in a free-style manner, moving from the initial assumptions and then focusing on its description, till proposing a way to quantify and measure the "legal risk" in a software project. Next table maps the structural elements used to the typical CMMI ones, with a short example and description.

Table 1. Mapping between [28][29] and CMMI PA Architecture elements [14]

Terminology [16][17]	CMMI PA Element(s) [18]	Description/Example
Steps	Specific Practices (SP)	For instance, the definition and analysis steps in [28] correspond to two SP for the Specific Goal #2 (SG2) about the project legal risk management along the whole Software Life Cycle (SLC) phases.
Risk source	Sub-practices	In [28][29] there is a list of risk sources, to be related to what CMMI calls "sub-practices" for a certain SP. For instance, User Requirement (UR) document, traceability document and prototypes have to be considered as sub-practice within SP2.2 (Definition).
Measures	Sub-practices General Practice (GP) 2.8	The list of risk measures proposed in [28][29] have been used as a source for refining the list of sub-practices within the process as well as the base for the Software Legal Assurance Percentage (SLAP) measure, to be used as a proposal in the 'Elaboration' section of GP2.8.

In the following sections the formulation for the LAS process is described, taking care to the two possible dimensions in a maturity model: the process and the capability ones.

[4] In the rest of the paper, we will refer to CMMI-DEV simply as "CMMI".

3.2 Process Dimension: Specific Goals & Specific Practices

Now it is time to reformulate in a CMMI-like style, starting with the process dimension, listing specific goals (SG) and specific practices (SP) by the content and ideas above presented.

3.2.1 SG1. Definition and Optimization of Business Opportunities

In current industrial environments, a major differential factor between companies is their ability to create and commercialize knowledge [10]. This is a strategic ability especially in the software market due to the intangible nature of software products and intellectual properties. As a result, a proper protection of these actives is a key element in the management of software organizations. Taking into account a strategy to manage intellectual property will generate and optimize business opportunities in the areas described in the next specific practices (SP).

SP1.1 Define the objectives for the Legal Assurance process
An inexistent or inadequate protection can reduce or even eliminate the commercial life of a software product. A successfully commercial product will generate clones that will be commercialized at a much lower cost avoiding development costs. It is therefore required to clearly define the objectives for a legal assurance process aligned with the organizational goals and policies. Objective of this practice is to verify the existence of a legal assurance process within the organization, in order to properly managed legal issues the organization has to deal with.

Sub-practices:
- product commercialization in internal and external markets;
- project funding.

SP1.2 Reduce or minimize risks for the Legal Assurance process
An adequate intellectual property protection will provide a powerful financial instrument that can be used to: guarantee credit applications; attract venture capital, or even and/or apply for government benefits and grants for Research and Development (R&D). As in the logic of a scorecard approach, part of financial resources obtained from the last fiscal year should be reinvested in the processes allocated in the so-called “Learning & Growth” perspective, related to innovation & infrastructure and people-related processes. Objective of this practice is the verification of the organizational capability to achieve this goal during time.

Sub-practices:
- determine available legal assurance activities;
- perform a descriptive analysis of previously defined activities;
- incorporate such activities into the software development lifecycle of the software product.

3.2.2 SG2. Management of Legal Risk through the Project Life Cycle

In addition to the possibility of maximizing business opportunities, assurance processes aim to reduce risks or potential threads derived from the failure to comply with the law or inadequate adaptation to legal regulations. Such issues, in turn, can generate legal claims from third parties, economic sanctions from governments or local authorities, and even penal actions that will obviously affect the successful outcome of a project. The problem of organizing legal assurance activities within the software development lifecycle is not a trivial process because each assurance activity must be applied at the right time, i.e., the efficiency of legal assurance activities depends on applying the right action at the right time. Therefore, it is not enough to perform legal audits or due diligence once the project has been completed. In the case of performing ordinary legal audits before launching a product, it is possible to find potential threads that force modifications of certain aspects of the project; usually such unplanned modifications are very expensive or cannot be performed. Neither sporadic actions by project managers to properly manage legal risks are enough.
In addition to the identification and incorporation of legal assurance activities at the right time, the management of these activities will be greatly improved if supported by a process that includes estimation techniques (from a quantitative point of view) about the protection level in a specific project.

SP2.1 Define the Legal Risk Management
First step is about the definition of the boundary of possible risks that could happen during the project lifetime. Moving from the project requirements, it is requested to find all possible explicit and implicit threats the project could express.

Sub-practices:
- verify the completeness and consistency of the UR document (i.e. to state clearly the features..., etc.);
- verify the completeness and consistency of the requirement traceability document;
- verify the completeness and consistency of the document about prototypes (i.e. the prototype is used only, ...etc)

SP2.2 Analyze the Legal Risk Management
After determining the list of possible project threats, they must be evaluated at the light of the people in charge of such rights and obligations, verifying the presence of mandatory clauses in contracts.

Sub-practices:
- establish the rights and obligations due by external personnel (i.e. establish with clarity the contractual figure...);
- establish the rights and obligations due by internal personnel (i.e. form by means of..., state in writing..., protect adequately...);
- verify the presence of mandatory legal clauses on the software development contracts (e.g. object for the contract, price, etc.)

SP2.3 Design the Legal Risk Management
Next step will be about the verification of collected elements against the applicable software licenses after the SRS is completed and before the Coding phase will start. This task represent an important step for quantifying the legal risk, if it would take place in the near future and therefore impacting on the project budget.

Sub-practices:
- verify and ensure the compliance to all the applicable software licenses;
- verify and ensure the protection of personal data.

SP2.4 Coding the Legal Risk Management
During the coding phase, it is requested an analysis of the source code produced by the organization, paying attention to the elements produced/incorporated within the software solution, as meta-tags or other DRM-related issues

Sub-practices:
- ensure the patrimonial rights for the product (e.g. finger-printing; watermarking techniques; introduction of innocuous and implausible code, etc.);
- verify the legal conformance of incorporated web elements (e.g. meta-tags).

SP2.5 Test and Release the Legal Risk Management
During the Test and Release phase, the accompanying actions requested on the legal side will face the duties for an eventual registration of the software product, according to kind of license established between the parties as well as the licensing documentation to be provided to the customer at the release.

Sub-practices:
- ensure the ownership of the product at the first release (i.e. register the product with IPO, notarial deposit, ...);
- write the licensing document and a template for the acceptance of the delivered product (according to the nature of the project: turnkey software development projects, customized software projects...).

SP2.6 Maintain the Legal Risk Management
Last but not least, the maintenance phase will require an update on a regularly basis of the legal assurance on the current contracts.

Sub-practices:
- write the maintenance contract;

- write the associated service level agreements (SLA);
- deposit to the notary of the project's content, graphical design, source code, and any other element identifying an improvement/evolution for such software.

3.3 Capability dimension: Generic Goals (GG) and Generic Practice (GP) elements

Within the capability dimension, the elements to take into account are Generic Goals (GG) and Generic Practices (GP). CMMI provides a general formulation for them (5 GG and 17 GP), with the same text, to be contextualized to each process area (PA) purpose and scope. In order to provide further guidance for implementation, in each PA there is often an informative model component, called "Elaboration", with tips on the "how" the GP should be applied uniquely to such PA.
Even if all GP could have an "Elaboration" section filled, moving from the initial proposal [28][29], measurement issues could – treated in GP2.8 (Monitor & Control the Process) - could have a particular benefit. In fact, the initial study had the main goal to quantify the legal risk to be managed, with the final result to provide a metric, called Software Legal Assurance Percentage (SLAP), with the objective to quantify the percentage measure of legal protection of a software product. The elements to take into account for determining the final percentage are the risk sources above mentioned. The average legal protection value from the four main SLC phases (planning, requirements, development and deployment) returns the final SLAP value.
Such value, those calculation process is explained with details in [28][29], could be inserted – jointly with its rationale – as a suggestion in the "Elaboration" section for GP2.8 of LAS process area. Next table summarizes the possible elaboration for LAS moving from the CMMI GP structure.

Table 2. GP Elaborations for LAS process

GG/ GP	Title	Elaboration for LAS
GG.1	**Achieve Specific Goals**	
1.1	Perform Base Practices	---
GG.2	**Istituzionalize a Managed Process**	
2.1	Establish an Organizational Policy	This policy establishes organizational expectations for establishing and maintaining guidelines about the typical legal elements to be verified for a certain software product (i.e. OSS, web portals, custom-based software, …)
2.2	Plan the Process	This plan for performing the legal assurance process can be included in (or referenced by) the risk plan, which is described in the Risk Management process area.
2.3	Provide Resources	Legal personnel should be – possibly – full time. Example of tools to be used is risk taxonomies as a source for analyzing possible legal implications.
2.4	Assign Responsibilities	---
2.5	Train People	Examples of training topics include the following: • Intellectual Property laws, Data Protection laws, Electronic Commerce and E-Marketing and E-Business regulations, Legal portion of Computer Programs: Copyrights and patents
2.6	Manage Configurations	Examples of work products placed under configuration management include the following: • Software development agreement, Outsourcing contracts, Taskforce contracts, Software Requirements Specification Contracts, Intellectual property registration
2.7	Identify and Involve Relevant Stakeholders	Examples of activities for stakeholder involvement include the following: • Technical People: Introduce Stenography Techniques: Fingerprint and watermarking. • Management people: Control Contracts and Agreements and negotiation documents
2.8	Monitor & Control The Process	Examples of measures used in legal assurance include the following • See [28][29] for quantifying the legal risk
2.9	Objectively Evaluate Adherence	Examples of activities reviewed include the following ones: • Monitoring and tracking project activities Examples of work products reviewed include the following ones: • Software Requirements Documents;

		• Code Components; • Mandatory Procedures for data protection regulations.
2.10	Review Status with H/L Mgmt	---
GG.3	**Istituzionalize a Defined Process**	
3.1	Establish a Defined Process	---
3.2	Collect Improvement Information	---
GG.4	**Istituzionalize a Quantitatively Managed Process**	
4.1	Establish Quantitatively Objectives for the Process	---
4.2	Stabilize Subprocess Performance	---
GG.5	**Istituzionalize an Optimizing Process**	
5.1	Ensure Continuous Process Improvement	---
5.2	Correct Root-Causes of Problems	---

3.4 Positioning LAS in the maturity model representations

Another long debate in the SPI community is about the choice between the staged and continuous representation: CMMI-DEV guide included a specific section in Part 1, presenting main 'pros & cons' about the two.
Looking at the *staged* representation, LAS is a process that should be performed and controlled from the early stages of a project; therefore our suggestion is to consider it as a ML2 or ML3 process. Because the strong impact of legal issues also for SME customizing an OSS under LGPL, as discussed in Section 1, it would be better to consider LAS as a basic process at ML2. A different positioning at ML3 could be argued by those considering the set-up of an internal legal staff (or the hiring of people with legal skills) within the organization as a higher maturity requirement.
Looking at the *continuous* representation in the CMMI-DEV model and the nature of LAS process, it must be included within the "Support" category. Table 2 shows the positioning of CMMI-DEV v1.2 processes by category and ML with the inclusion of LAS.

Table 3. CMMI process areas by Process Categories and Maturity Levels

Process Categories *ML*	**Process Management**	**Project Management**	**Engineering**	**Support**
Optimizing	OID			CAR
Predictable	OPP	QPM		
Defined	OPF OPD OT	IPM RSKM	RD TS PI VAL VER	DAR
Managed		PP PMC SAM	RM	**LAS** CM MEA PPQA
Initial	*Ad-hoc processes*			

4 Conclusions & Prospects

For financial, commercial or product quality reasons, a suitable legal assurance process is a management aspect that cannot be ignored by the organizations developing information systems or subcontracting such services. Independently from the kind of software projects and licenses managed (open source, customized software, inclusion of web services, etc.), there is a need to adequately manage issues related to intellectual property and digital rights.
The results presented in this work have the objective of improving the professional practice of legal risks management for the information systems industry. To do so, it is needed to analyze systematically all legal assurance activities and measures that can jeopardize a project and align them with current software process assessment and improvement models.
With this aim, a series of transversal activities to be included within the software lifecycle process have been described. To structure the legal assurance activities as a process model, CMMI architecture

has been considered as the starting point. Finally, we also presented the main goals and practice for a new legal audit process has been proposed.
Next work will be spent on the complete drawing of such process area, in order to be proposed as Change Request (CR)[5] to the Software Engineering Institute (SEI).

5 Literature

[1] Rejas-Muslera R.J., Cuadrado-Gallego J.J., Rodríguez D., *Defining a Legal Risk Management Strategy: Process, Legal Risk and Lifecycle*, in Proceedings of the 14th European Conference on Software Process Improvement (EuroSPI 2007), Potsdam, Germany, September 26-28, 2007, Lecture Notes in Computer Science 4764 Springer 2007, pp. 118-123 , ISBN 978-3-540-74765-9

[2] Cearley, D.W., Fenn, J., Plummer, D.C., Gartner's Positions on the Five Hottest IT Topics and Trends in 2005, Gartner, Publication Id. G00125868, 12 May 2005, URL: www.gartner.com/DisplayDocument?doc_cd=125868 {2008-05-18}

[3] FSF, Free Software Foundation – Licenses, URL: www.fsf.org/licensing/licenses/ {2008-05-18}

[4] Rejas-Muslera, R.J., Cuadrado-Gallego, J.J., Dolado, J., Rodríguez, D., The open source software vs. proprietary software debate and its impact on technological innovation. *Upgrade* [Online]. *6(5)*, October 2005, pp.35-39. ISSN: 1684-5285. URL: www.upgrade-cepis.org/issues/2005/5/upgrade-vol-VI-5.html {2008-05-18}

[5] Olliance Group, 2007 Open Source Think Tank: The Future of Commercial Open Source, Executive Summary Report, URL: thinktank.olliancegroup.com/ostt2007report.pdf {2008-05-18}

[6] Urban J., Legal Uncertainty in Free and Open Source Software and the Political Response, from "The Politics of Open Source Adoption (POSA), version 1.0", Social Science Research Council (SSRC), Eds: Karaganis J. & Latharn R., May 2005, pp. 68-82, URL: http://www.ssrc.org/wiki/posa/ {2008-05-18}

[7] Haislmaier J., Hamley C. & Cohn A., Open Source License Enforcement Actions. What you can expect when there is a knock on you door, May 23 2007, Presentation, URL: www.hro.com/resources/custom/publications/HRO%20Publications/opensourceppt.pdf {2008-05-18}

[8] Moyle K., Total Cost of Ownership and Open Source Software, Research Paper, July 2004, Dept. of Education and Children's Services (DECS), Government of South Australia, URL: www.curriculum.edu.au/verve/_resources/total_cost_op.pdf {2008-05-18}

[9] Tazi O., A Practical Strategy for Leveraging Open Source, Presentation, ObjectWebCon2006, 5th ObjectWeb Annual Conference, January 31 February 2 2006, Paris (France), URL: objectwebcon06.objectweb.org/xwiki/bin/download/Main/DetailedSession/O-Tazi.pdf {2008-05-18}

[10] WIPO, Records of the Intellectual Property Conference of Stockholm (Stockholm, June 11 to July 14, 1967), Volume II, pp. 283-330, URL: http://www.oup.com/uk/booksites/content/9780198259466/15550029 {2008-05-18}

[11] Sarbanes-Oxley Act, January 23 2002, URL: http://news.findlaw.com/hdocs/docs/gwbush/sarbanesoxley072302.pdf {2008-05-18}

[12] Janssens, L., Auditing CMMI Maturity and Sarbanes-Oxley Compliance, ISACA Journal Online, 2007, Vol.3, URL: www.qpit.ltd.uk/LinkedDocuments/ISACA%20Journal.pdf {2008-05-18}

[13] Tower, J., IB Technology Examples of CMMI Benefits, JP Morgan, version v1.4, August 2004, URL: www.sei.cmu.edu/activities/cmmi/results/pdfs/2004-CMMI-020.pdf {2008-05-18}

[14] CMMI Architecture Team, Introduction to the Architecture of the CMMI Framework, Technical Note, CMU/SEI-2007-TN-009, Software Engineering Institute, July 2007, URL: www.sei.cmu.edu/publications/documents/07.reports/07tn009.html {2008-05-18}

[15] CMMI Product Team, CMMI for Development, Version 1.2, CMMI-DEV v1.2, CMU/SEI-2006-TR-008, Technical Report, Software Engineering Institute, August 2006, URL: www.sei.cmu.edu/publications/documents/06.reports/06tr008.html {2008-05-18}

[16] Copeland L., The Maturity Maturity Model™ (M3), Stickyminds, URL: http://www.stickyminds.com/se/S6653.asp {2008-05-18}

[17] INCOSE, Risk Management Maturity Level Development, RMRP 2002-02, Version 1.0, April 2002, URL: www.pmi-switzerland.ch/fall05/riskmm.pdf {2008-05-18}

[18] Crosby, P., Quality is free, Mc-Graw Hill, 1979, ISBN 0070145121 (see also: http://c2.com/cgi/wiki?QualityManagementMaturityGrid {2008-05-18})

[19] RIMS, RIMS Risk Maturity Model (RMM) for Enterprise Risk Management, November 27 2006, URL: www.rims.org/RMM {2008-05-18}

[20] IACCM, IACCM Capability Maturity Model, August 2007, URL: http://www.iaccm.com/maturity/index.php {2008-05-18}

[21] OGC, Portfolio, Programme & Project Management Maturity Model (P3M3), version 1.0, February 1, 2006, URL: www.ogc.gov.uk/documents/p3m3.pdf

[5] www.sei.cmu.edu/cmmi/models/change-requests.html.

[22] OGC, Prince2 Maturity Model (P2MM), v1.0, March 2006, URL: www.ogc.gov.uk/documents/PRINCE2_Maturity_Model_Version_1.pdf {2008-05-18}
[23] PMI, Organization Project Management Maturity Model (OPM3), URL: http://opm3online.pmi.org/ {2008-05-18}
[24] ISO/IEC, IS 12207:1995 – Information Technology – Software Lifecycle Processes, International Organization for Standardization, Genève, July 20 1995
[25] ISO/IEC, IS 15288:2002 – Systems Engineering – System Life Cycle Processes, International Organization for Standardization, Genève, November 11, 2002
[26] ISO/IEC, IS 15504-x - Information Technology - Software Process Assessment, Parts 1-6, 2003-2007, URL: www.isospice.com {2008-05-18}
[27] CMMI PRODUCT TEAM, *CMMI for Acquisition,* Version 1.2, CMMI-ACQ v1.2, CMU/SEI-2007-TR-017, Technical Report, Software Engineering Institute, November 2007, URL: www.sei.cmu.edu/publications/documents/07.reports/07tr017.html {2008-05-18}
[28] Rejas-Muslera, R., Proceso De Gestión De Riesgos Legales Para Proyectos De Desarrollo De Software, Ph.D Thesis, Universidad de Alcalá, Departamento de Ciencias de la Computación, Madrid (Spain), May 2007
[29] Rejas-Muslera, R.J., Cuadrado-Gallego, J.J., Sicilia, M.A., Rodríguez, D., SLA: A legal assurance process model for software engineering management. Software Process: Improvement and Practice (SPIP), Wiley & Sons, Vol.12, Issue 2, March-April 2007, pp.191-198

6 Appendix A: List of Acronyms

ACQ	Acquisition
AM	Agreement Management
ARD	Acquisition Requirements Development
ATM	Acquisition Technical Management
AVAL	Acquisition Validation
AVER	Acquisition Verification
BSD	Berkley Software Distribution
CMMI	Capability Maturity Model Integration
CMMI-ACQ	CMMI for Acquisition
CMMI-DEV	CMMI for Development
CR	Change Request
DRM	Digital Rights Management
FSF	Free Software Foundation
GNU	GNU's not Unix
GP	General Practice
GPL	General Public License
ICT	Information & Communication Technology
LAS	Legal Assurance process
LGPL	Lesser GPL
MIT	Massachusetts Institute of Technology
ML	Maturity Level
MM	Maturity Model
OPM3	Organizational Project Management Maturity Model
OSS	Open Source Software
P2MM	Prince2 Maturity Model
P3M3	Portfolio, Programme and Project Management Maturity Model
PA	Process Area
PMMM	Project Management Maturity Model
QMMG	Quality Management Maturity Grid
RD	Requirement Development
RMMM	Risk Management Maturity Model
ROI	Return On Investment
RSKM	Risk Management
SAM	Supplier Agreement Management
SEI	Software Engineering Institute
SG	Specific Goal
SLAP	Software Legal Assurance Percentage
SLC	Software Life Cycle
SME	Small-Medium Enterprise

SOX	Sarbanes-Oxley
SP	Specific Practice
SPICE	Software Process Improvement Capability dEtermination
TCO	Total Cost of Ownership
UR	User Requirement

7 Author CVs

Luigi Buglione

Dr. **Luigi Buglione** is an Associate Professor at the École de Technologie Supérieure (ETS) – Université du Québec, Canada and is currently working as a Quality & Process Engineer in the Quality Dept. at Engineering.IT (formerly Atos Origin Italy) in Rome, Italy. Previously, he worked as a Software Process Engineer at the European Software Institute (ESI) in Bilbao, Spain, and Sema Group in Rome (Italy). Dr. Buglione is a regular speaker at international Conferences on Software Measurement and Quality, is part of the Board of Directors of the Italian Software Metrics Association (GUFPI-ISMA), where coordinates the Software Measurement Committee (SMC).
He developed and was part of ESPRIT and of Basque Government projects on metric programs, EFQM models, the Balanced IT Scorecard and QFD for software and was a reviewer of the SWEBOK project. He received a Ph.D in Management Information Systems from LUISS Guido Carli University (Rome, Italy) and a degree in Economics from the University of Rome "La Sapienza", Italy. He is a Certified Software Measurement Specialist (IFPUG CSMS) Level 3. He can be contacted at luigi.buglione@eng.it or luigi.buglione@computer.org.
Further info on the SEMQ website: www.geocities.com/lbu_measure/

Ricardo J. Rejas-Muslera

Dr. **Ricardo J. Rejas-Muslera** received his Ph.D in Computer Science from the Alcalá University of Madrid, and his B.S. degree in Law from the Carlos III University of Madrid; also he is Master in Law (Complutense University of Madrid).
He worked at American Telecom as legal advisor from 1999, in 2002. In the fall of 2005 he moved at the University of Francisco de Vitoria, Madrid, Spain where he is full Professor, and since 2007, assistant director at the Engineering Department. In addition since 2005 he has been professor in Master Program at the Alcalá University of Madrid.
Dr. Rejas-Muslera research interests include different aspects of Information Technology especially software legal aspects. He is the Head of the Integra Research Unit at the University of Francisco de Vitoria, and since 2006, is the head of the 6SPIN, an integrant part of the Software Process Improvement Network (Software Engineering Institute, University of Carnegie Mellon). E-mail: r.rejas.prof@ufv.es

Juan José Cuadrado Gallego

Juan J. Cuadrado-Gallego is Profesor Titular de Universidad and the Director of the Ph.D. degree in Computer Science at the Computer Science Department of the Universidad de Alcalá, Madrid, Spain; he is also Consultor in the Universitat Oberta de Catalunya and has been Teaching Staff in the Università degli Studi Roma Tre, Roma, Italy, in 2004, and in the Otto-von-Guericke Universität, Magdeburg, Germany, in 2008.
Dr. Cuadrado-Gallego is Chief Editor of the Journal of Software Measurement, ISSN:1862-7420, and Permanent Member of the Advisory Board of the International Conference on Sofware Process and Product Measurement, Mensura. Since 2006 he is the President of the Spanish Function Points Users Group (SFPUG), official chapter in Spain of the International Function Points Users Group (IFPUG), and since 2007 he is Member of the International Advisory Commitee of the COmmon Software Measurement Consortiun (COSMIC). E-mail: jjcg@uah.es

SPICE Experiences with Management Processes

Gerhard Lichtenecker, Augustin Trücher
Magna Steyr Fahrzeugtechnik AG & Co KG
gerhard.lichtenecker@magnasteyr.com augustin.truecher@magnasteyr.com

Abstract

This paper gives an overview about the method "Process Assessment for Management Processes", developed at MAGNA STEYR Fahrzeugtechnik based on the SPICE philosophy to improve the effect of internal audits on the continuous improvement of the organisation.

Keywords

Automotive Excellence

Audit

Continuous Improvement

ISO TS 16949

Management Processes

Management System

Process Assessment

Process Improvement

SPICE

1 Introduction to MAGNA STEYR

MAGNA STEYR is the leading global, brand-independent engineering and manufacturing partner for the automotive industry. The company offers OEMs solutions for a wide range of services with highly flexible development and assembly strategies. The range of services comprises the whole range of processes within the automotive industry, from individual engineering services to complete vehicles, and from extra-low volume through peak shaving to volume production. MAGNA STEYR has about 10.000 employees at 10 locations world wide.

2 Management system at MAGNA STEYR

MAGNA STEYR does not only aim to meet their customers requirements, but also those of their employees, owners, suppliers and society while taking into consideration cost effectiveness, sustainability, quality, environmental protection, occupational health and safety and data protection at the same time. To reach these targets, MAGNA STEYR has introduced an integrated management system.

2.1 Quality management

A finished product's quality also depends on the manufacturer's quality. MAGNA STEYR was the first complete vehicle manufacturer in the world to be certified to ISO/TS 16949:2002.

2.2 Environmental protection and occupational health and safety

The efforts in this field are based on MAGNA International's health, safety and environmental policy and the MAGNA Employee's Charter.

The company's commitment to the environment starts with researching into and developing forward-looking, recyclable products and processes. Examples of these are the development of a hydrogen tank for passenger vehicles or the use of water-based paints in vehicle assembly.

2.3 Information security

MAGNA STEYR employees play a significant role in enabling to offer customers innovative solutions and products of an excellent quality at a competitive price.

Not only the employees' expertise goes into these innovative solutions and products, but also that of customers and partners. Customers and partners therefore rely on their data, knowledge and information being treated confidentially and securely at MAGNA STEYR.

Information security as practised at MAGNA STEYR is part of the success, borne out by the confidence placed by renowned customers. Confidence is a firmly rooted part of the integrated management policy.

Every employee at MAGNA STEYR commits to maintain strict secrecy regarding all business and trade secrets (e.g. manufacturing processes, working methods, plant and equipment, projects, innovations, design drawings, etc.). It is regardless whether such information is available on paper, in electronic form (as e-mail or in systems), as a photo or film or disclosed verbally (by telephone/at a meeting).

2.4 Business process model

The business process model of MAGNA STEYR is based on the integrated management system.

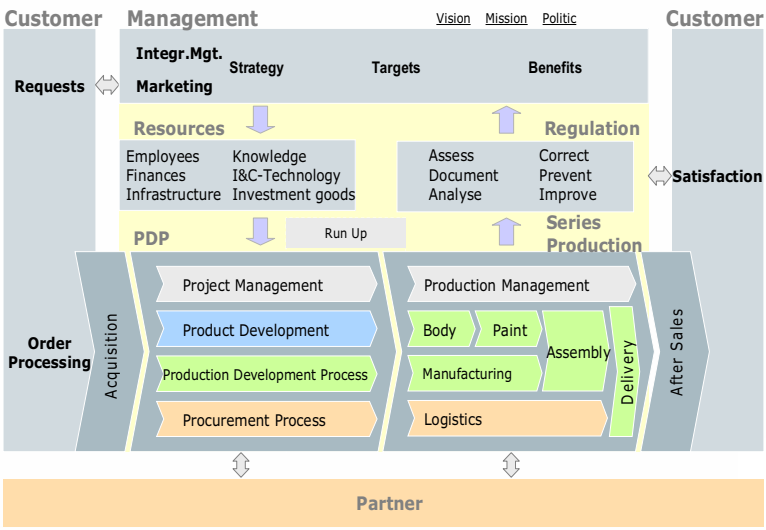


Figure 1: Business process model of MAGNA STEYR

At the top level 16 business processes have been defined:

1. Company management
2. Marketing and sales
3. Finance management
4. Employee related processes
5. Knowledge management
6. IT- management
7. Infrastructure management
8. Product development process
9. Development of production processes
10. Purchase processes
11. Production Management
12. Manufacturing of components
13. Parts supply (Logistics)
14. Test equipment monitoring
15. Technical analysis of materials
16. Guarantee and field claims

For each of this top level processes a process owner is assigned and is responsible for the overall performance of the process within MAGNA STEYR. In each business unit a process manager is installed to control the process within his unit. Furthermore each of this top level processes is split up into sub- processes which describe the activities in more detail.

3 Certificates and audits

3.1 General

MAGNA STEYR is certified according to several international standards.

- Environmental protection: EU – Regulation EMAS II 761/2001 and ISO 14001:2004
- Occupational health and safety: OHSHS 18001:1999
- Information security: ISO/IEC 27001:2005
- Quality management system: ISO/TS 16949:2002

To maintain the certifications, internal and external audits are necessary. The requirements of all above mentioned standards are checked together during integrated audits to save time and cost and to increase effectiveness.

The aim of the internal audit system is to support the consequent operation and improvement of the integrated management system and to assure the compliance with the normative and legal terms. The results of the internal system audit in the form of findings and positive and negative statements are part of the management assessment by the top management.

The internal audits are co-ordinated by the audit- group within the functional department “Quality Management” who plans the audit program for an audit year in co-ordination with all system representatives and the process owners of the integrated management system. The audit year extends from the period of 1st September to 31st August of the following year with the external audit as a final step.

To conduct the internal audits, a pool of auditors is used, who are working in different divisions and departments of MAGNA STEYR. Each member of this pool should conduct one internal audit per year on average. This supports information sharing throughout the organisation.

3.2 The limitations of audits

Besides the internal audit, which is done once a year for each process and its sub- processes, the process owners have to make a process review regularly where the process design is checked and the key performance indicators of each process are compared with the targets.

As the performance of the processes is assessed and corrective / improvement actions are defined at the process review, the process owners and process managers acknowledge the internal audit as an additional exam, with the threat to fail.

The added value of an internal audit decreases with increasing maturity of the processes, as the processes are not benchmarked with the best in class but with the minimum requirements of the standards. The added value can be measured by the number of nonconformities discovered during the audit.

Conducting the internal audits for several years, it is obvious, that the number of nonconformities decreases from year to year (see figure 2)

The aim of continuous improvement of processes is not sufficiently supported by the standards (see figure 4). Based on these facts, MAGNA STEYR decided to develop a process assessment method, which has to

- fulfil the requirements of the standards for internal audits and to
- support the continuous improvement of the business processes.

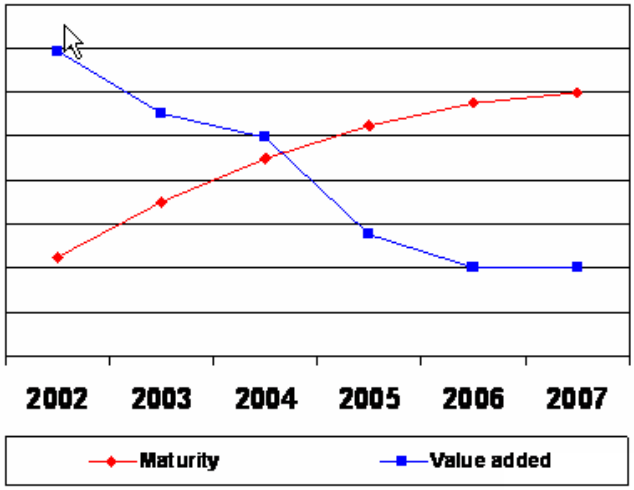


Figure 2: value added at an internal audit versus maturity of the process

3.3 A new method: process assessment

After the analysis of different assessment models MAGNA STEYR decided to use a system similar to Automotive SPICE as basis for the process assessments. Automotive SPICE was developed for the evaluation of software and system development processes of automotive suppliers in accordance with ISO/IEC15504. However the systematic behind Automotive SPICE is applicable to all kinds of processes due to the neutral logic.

The MAGNA STEYR assessment model comprises - besides the generic practices, which are similar to those in the SPICE assessment model - requirements of the automotive quality standard (ISO TS 16949:2002), requirements for occupational health and safety (OHSAS 18001:1999), information security (ISO/IEC 27001:2005), environmental protection (EU – Regulation EMAS II 761/2001 and ISO 14001:2004), requirements based on the automotive excellence (VDA 18.1) and process specific base practices.

In January 2007 a team started with the development of the MAGNA STEYR assessment model and the procedures for the assessments. A concept was developed how to integrate the requirements of internal audits and SPICE into one system. In November 2007 the first process assessment was conducted on the process “production management”. After improvement of the proceedings based on lessons learned the “MAGNA STEYR Process Assessment” was implemented as a standard method in 2008 and can be used instead of internal audits. The overall rollout is planned to be completed at the end of 2009

The first assessment for each process is used to define the status quo – no targets are set. The results of the assessment are discussed with the process owner and the top management and targets for the following assessment are defined.

One of the main differences between the process assessment and the audit is the fact that the assessor has the role of a facilitator during the assessment workshop. The assessor has the task to support the process owner and the process managers with tools like process analysis and SWOT- analysis.

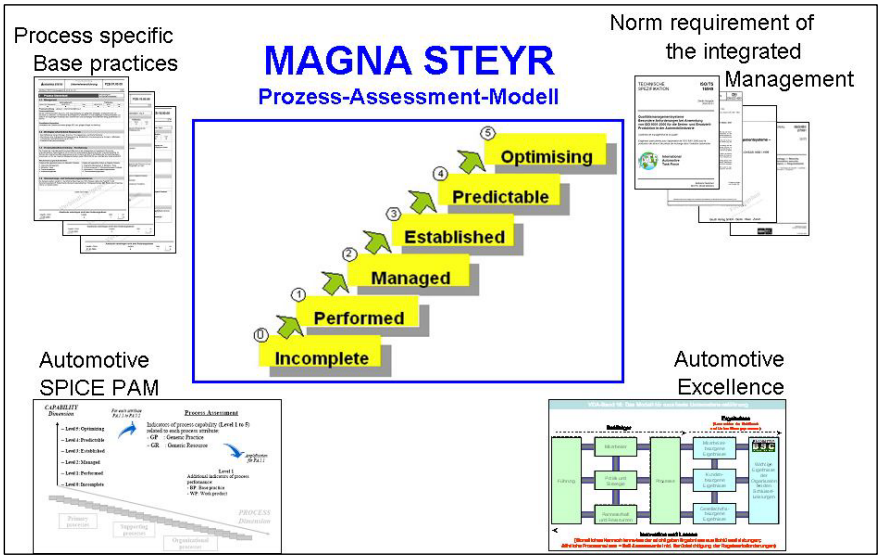


Figure 3: inputs for the MAGNA STEYR process assessment model

As mentioned above, the process assessment model is a combination of different requirements. As an example the assessment model for the production management process is shown.

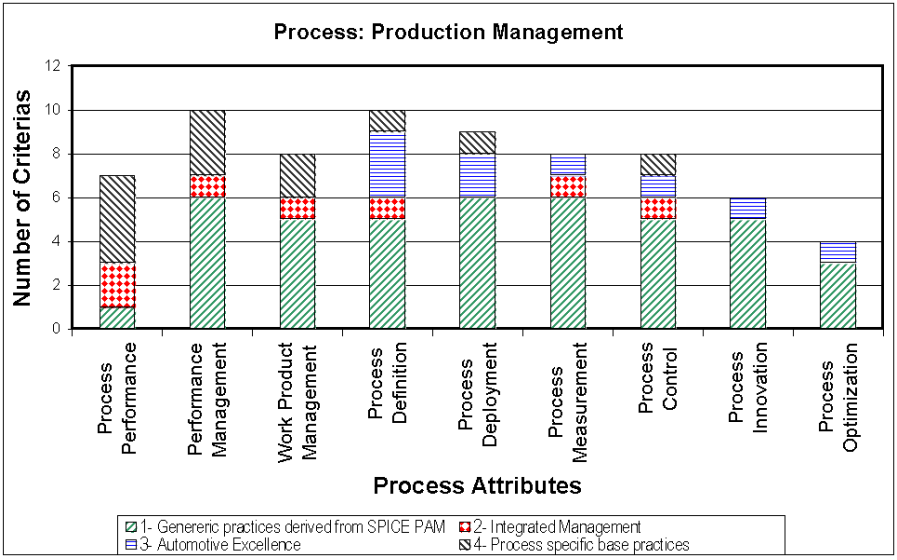


Figure 4: MAGNA STEYR Process Assessment model, example production management

To support the documentation of the assessments MAGNA STEYR's audit support tool, a Microsoft Access based database was used and extended with the contents of the process assessments.

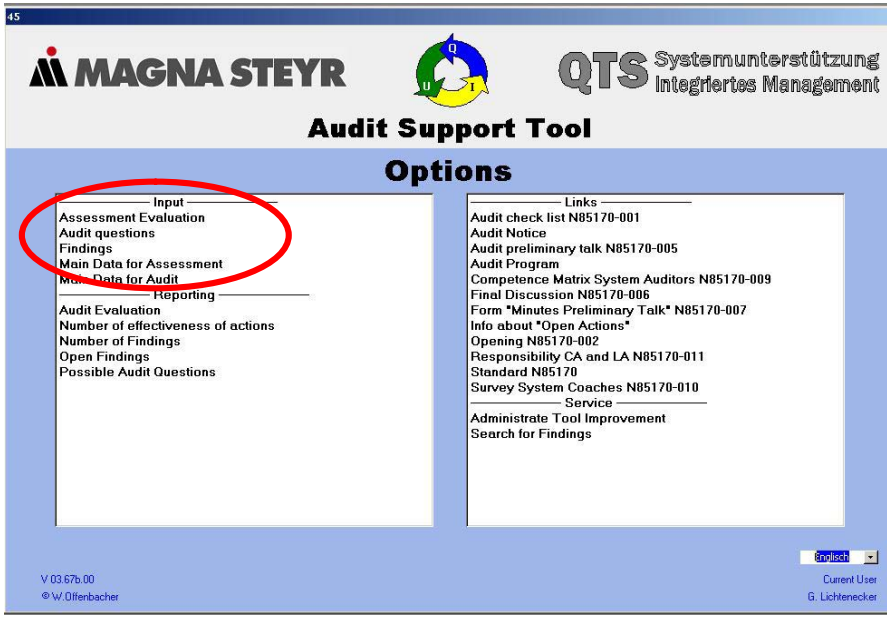


Figure 5: MAGNA STEYR's audit support tool with the extension for process assessments

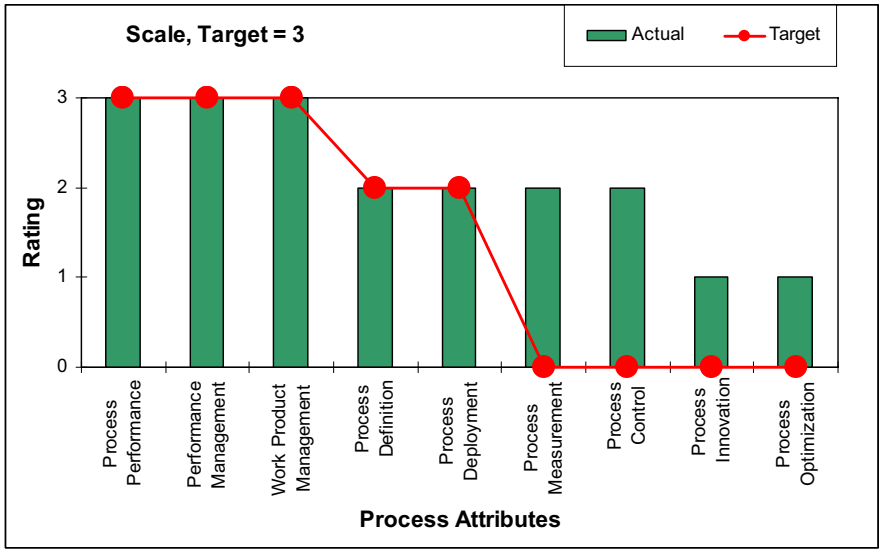
3.4 First results

Before starting with the first pilot assessment with the process "production management" in November 2007 assessors have been nominated and trained. The assessment started with a self evaluation of the process owner, followed by an assessment workshop. The process level was rated with „established".

Based on the positive feedback from the process owner, the top management decided to establish the MAGNA STEYR process assessment as a standardized method. Two more process assessments were conducted on the processes "infrastructure management" and "marketing and sales", the rollout at MAGNA STEYR Graz is planned to be completed at the end of 2009.

Some highlights of the positive feedback:

- Assessment in form of a workshop with the assessor as facilitator is more effective than a traditional audit.
- The use of analysis tools helps to identify process strengths and - potential by means of SWOT analysis.
- Additional process reviews are not necessary any more.
- The results are presented in form of ratings on a scale.
- Process owner and managers are involved in the analysis of the process.
- See the potential for the next level



3: fully achieved, 2: largely achieved, 1: partially achieved, 0: not achieved

Figure 5: assessment results for the pilot assessment (process: production management)

The assessment also showed things that should be improved:

- One of the key performance indicators of the process does not support the aim to control the process.
- Some measures to control the quality of the products, which are planned to be implemented during the serial production, should be considered in the Product development phase.

4 Summary

At MAGNA STEYR Fahrzeugtechnik in Graz, Austria the method “MAGNA STEYR Process Assessment” was developed, to combine the requirements of internal audits and continuous improvement into a powerful, efficient and effective tool.

The combination of the SPICE philosophy with the business excellence model, normative requirements and process specific criteria helped to define a way, how process assessment can achieve maximum value added.

5 Literature

ISO/IEC 15504-ff

VDA 18-ff

VDA manual “Automotive Spicec“

ISO/TS 16949

EU – Regulation EMAS II 761/2001

ISO 14001:2004

OHSHS 18001:1999

ISO/IEC 27001:2005

Author CVs

Gerhard Lichtenecker

Gerhard Lichtenecker – Head of Quality Advanced Development at MAGNA STEYR Fahrzeugtechnik in Graz, Austria.

The main focus is on the development and introduction of quality methods, supporting fault prevention and process improvement.

Mr. Lichtenecker has a Master degree in electrical engineering at the technical university in Graz, Austria; has been awarded with the Six Sigma Black Belt certificate (issued in 1999 by the University of Texas at Austin in cooperation with Air Academy Associates) and since 2007 holds the certificate for DFSS Master Black Belt issued by Air Academy Associates, Colorado Springs. He is Quality Systems Manager and Quality Auditor.

In previous assignments he was head of engineering risk management at MAGNA STEYR Fahrzeugtechnik in Graz and before joining MAGNA he was responsible for the Six Sigma Program at VA-Tech Hydro (power plant and generator company in Austria) where he closely cooperated with General Electric, one of the key customers.

Augustin Trücher

Augustin Trücher is a member of the quality advance development department at MAGNA STEYR Fahrzeugtechnik in Graz. His main focus is the development and introduction of quality methods with special emphasis on process management and TQM. He is Quality Systems Manager and Quality Auditor.
Before joining MAGNA he was responsible for the Statistical Process Control and Gage Control at Philips Components Austria and Quality Manager at BAUER-Industry, a supplier for the automotive industry.

Change Management a Key Factor for Improve the Process Deployment: A case study

Bayona Oré Sussy, Calvo-Manzano Jose Antonio, Cuevas Gonzalo, San Feliu Tomás
Languages and Informatics Systems and Software Engineering Department
Faculty of Computer Science, Polytechnic University of Madrid
Campus Montegancedo, 28660 Boadilla del Monte Madrid, Spain.
sbayona@zipi.fi.upm.es, jacalvo@fi.upm.es, gcuevas@fi.upm.es, tsanfe@fi.upm.es

Abstract

This paper describes a case study on process deployment in five software development and maintenance sites. It focuses on the Process Asset Library and the defined process as key elements for Process Deployment. A clear understanding of change management is necessary to ensure the adoption and institutionalization of new processes. In order to guarantee process deployment efficiently, strategies that include training and communication are necessary. Finally the use and acceptance level of processes have been considered in the case study.

Keywords

Process Deployment, Process improvement, Change management, CMMI

1 Introduction

Nowadays, it is necessary that software factories and software development and maintenance centers give response to the client's demands in a competitive environment by producing products and services of quality, basically, in terms of time and costs. Methodologies and processes model have been developed to help the organizations to achieve these objectives. One of the most used was the "Capability Maturity Model" (CMM) that has just evolved into CMMI, "Capability Maturity Model Integration" (CMMI) [2]. The Software Engineering Institute (SEI) developed the CMMI models to integrate improvement efforts.

The current CMMI for Development (CMMI-DEV) model has 22 process areas. CMMI-DEV is centered in the maturity of the organization and it has five levels: initial, managed, defined, quantitatively managed and optimizing. Each maturity level defines a set of process areas. A CMMI maturity level 3 organization has a defined and measured process library called Process Asset Library (PAL). The PAL contains the standard processes, the defined processes, and the tailoring guidelines among others elements. That is, the process is established, documented and measured [9].

Many software factories have decided to follow this model motivated by the necessity of being certified at a certain level [14] and of improving the quality of their processes. In many cases it is a client requirement or it is simply an advantage when competing for a contract [12].

The Software Processes Improvement Group is the most important component of the improvement infrastructure and it is in charge of the standard process definition. The defined processes and the tailoring guidelines are part of the PAL. Garcia said "The PAL is an organized, well indexed, searchable repository of process assets that is easily accessible by anyone who needs process guidance information, data files, templates or other support materials" [6].

Once processes are defined, they must be deployed and institutionalized in the organization in order to get people to use the new processes. However, the adoption and institutionalization of new processes is difficult because they are conditioned by the relationships that exist between people and artifacts (methods, tools, processes and paradigms) to be deployed.

A change management approach is necessary to ensure that the staff works as expected. Change is an inevitable consequence when implementing new or updating processes. Process improvement depends on the capacities of the organization (human factors, financial resources and materials) and the business objectives. If the staff is not involved in process definition and the detail of contents is not appropriate change resistance will increase.

A PAL's effectiveness will be diminished if personnel are not sufficiently trained. Not providing training might lead to misuse and/or negligence [6]. Some studies show that the lack of communication, training, defined roles of the organization, change management, motivation can cause failure during the software product development [7] [11].

This paper shows the results of a research work [1] which describes the deployment experience in five software development and maintenance sites and focuses on the processes deployed, the methodology used by each site and the deployed processes status.

The rest of the paper is organized as follows: section 2 has a brief introduction to the process deployment. Section 3 presents the methodology used to carry out the research of deployment process status in multisite organizations and the survey results and Section 4 presents conclusions and future research work.

2 Process Deployment

The purpose of process deployment is about getting people to use the new processes. It is frequent that an organization uses the term *process implementation* and not *process deployment*. There is a difference between the implementation and deployment concepts. International Process Research Consortium (IRPC) [16] indicates that "It is important to recognize the critical difference between process implementation and process deployment. The concept of deployment goes beyond the single instantiation of an implemented process, to address the effective deployment of a process specification to achieve multiple implementations across an organization, each tailored to suit its specific organizational context".

Intensive research into the human factor and change management is needed (handling resistance to change). For this reason the IPRC has included the topic of Process Deployment in a list of research items for the next 10 years.
The process deployment is focused on people and it incorporates aspects like training to develop collective abilities, staff motivation and effective communication.
According to our research work the process deployment elements are:

- **The organization**. Is the place where the new processes will be deployed. The organizational and functional structure that is established in each site is part of this element. In this study the sites are identified as site 1, site 2, site 3, site 4 and site 5. Site1, site 2 and site 3 are software maintenance sites. Sites 4 and 5 are software factories.
- ***The Process Asset Library***. Is constituted by the defined processes and the tailoring guidelines for each process area. The organization's set of standard processes are supported by the Organizational Process Definition (OPD) and Organizational Process Focus (OPF) process areas. The purpose of OPD is to establish and maintain a usable set of organizational process assets and work environment standards. The purpose of OPF is to plan, implement and deploy organizational process improvements based on a thorough understanding of the current strengths and weaknesses of the organization's processes and process assets [2].
- ***The process.*** A process is a goal-directed, interrelated series of actions, events, mechanisms, or steps that are constituted by (i) the roles and responsibilities of people in order to carry out their work, (ii) the tools and equipment that allow them to develop their work and (iii) the procedures and methods that define how to carry out the tasks and their relationship. The processes definition should be based on organizational needs, experiences and lessons learned. However, it is a task which must be approached with great care because: (1) the knowledge must be elicited to identify the best practices, (2) weaknesses in processes have to be assessed and (3) process description has to be stored [13]. Also, the people must be convinced that processes will improve quality and achieve benefits [4].
- ***The human resources.*** With the required knowledge, skills and attitudes tasks can be carried out satisfactorily and thereby ensure that all activities are completed. The change is related to process deployment. Then, if the staff resistance to change is not negotiated, processes adoption and institutionalization will be more difficult. Aspects like people capacity and competence, teamwork, alignment to the organizational vision, process improvement proposals, participation of those involved [10], training and communication are all key factors to develop human resources [3]
- ***The process of process deployment.*** Is focused on people at all levels of analysis: individuals, teams, groups, organizations, countries and cultures. At the organizational level, it is the character of people monitoring and putting processes into practice in their daily work that has to be controlled [5]. It consists on selecting a process deployment strategy where information, communication, training and evaluation are the principal items of this strategy. An important aspect is the people's disposition to continue and apply the processes to their daily work, in such a way to assure that the processes are continued.
- ***The process deployment methodology.*** Establishes the guidelines, procedures and the rules to deploy the processes. It contains the activities that should be developed for training and motivating people in the new processes and the communication strategies. Also, it establishes deployment roles and responsibilities. The applicability of the methodology depends on several factors such as: the complexity of the projects [15], the knowledge of those involved.
- ***The change management*** is a key factor to minimize the staff resistance to the new processes [8]. The change management strategies should satisfy the need of people to feel sure of their knowledge and competence and at the same time their need for change which may boost their opportunities and professional expectations.

Figure 1 shows the Process Deployment elements.

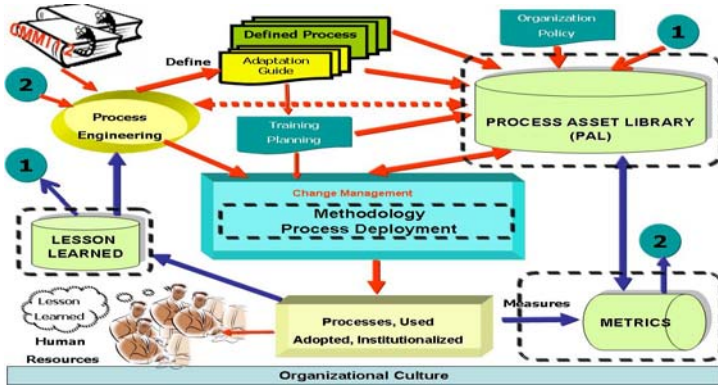


Fig. 1 Process Deployment elements

3 Research Methodology and Results

The objective of this study was to analyze the process of Process Deployment in each of the selected sites. The deployed processes, the methodology used and the deployed processes status (use and acceptation level) are analyzed. Other objective of this study was to identify the main factors that contribute to processes institutionalization.

The following activities were carried out to achieve these objectives:

- ***Identify the issues to research and develop the work plan.*** The organizational structure, the human resources and those involved are included in this activity. Based on the previous research work the potential success factors of process deployment were identified. The process status was evaluated in three critical aspects: use, acceptance and complexity level.
- ***Identify those in charge of the process implementation at the sites.*** The people in charge of process deployment were identified. The study has considered *individual experience* in process deployment.
- ***Identify the organizational processes.*** Current documentation in each site, the Process Asset Library, the implemented processes, the standard tools and forms were analysed. In the case of the maintenance sites the quality group was in charge of the process definition. The process definition was in charge of quality group of site 1. Those processes were deployed in site 2 and 3. In the software factories, site 5 was in charge of process definition and these processes were deployed in site 4 with adjustments.
- ***Develop a survey with open and close questions.*** This survey was divided into the following modules: (1) Module I, questions related to the organization, staff and organizational structure, (2) Module II, questions related to the deployed processes and their status and (3) Module III, questions related to the methodology used for the deployment.
- ***Fill in the survey by the personnel in charge of implementing the processes.*** The survey was answered by the people in charge of the process deployment.

The survey was carried out in five sites. Each site established its own procedures to carry out the implementation of the processes.

The information analysis was carried out taking into account the following aspects: (1) the Process Asset Library that includes the deployed process, (2) the deployment methodology used and (3) the impact on the use and adoption of processes.

3.1 The Process Asset Library

The sites have a Process Asset Library (PAL). The PAL is constituted by the defined processes established according to the organizational needs and it was defined by the Quality Committee.

The sites identified the key processes and how they interact. This PAL was deployed in groups of processes according to the priority criteria established by the sites. Initially, Site 1 and Site 5 were in charge of defining the processes following the CMMI model. In Site 1, processes were defined by people there and, then the processes were deployed to Site 2 and Site 3. Site 5 processes were deployed to Site 4. Site 4 tailored these processes to obtain a CMMI certification.
To deploy processes, they were grouped in the following categories: operative processes, monitoring and control processes, support management processes and quality assurance processes.
Each defined process has support tools, forms, guidelines and metrics. The metrics will allow the audit and control of process performance by the Quality Committee.
Figure 2 shows the processes deployed at the sites.

Fig 2. Processes deployed in sites

3.2 Deployment Process Methodology

Each site defined its own deployment process to deploy their processes. In each site, the deployment process objective was to manage the adoption and use of the new processes.
Table 1 shows the general steps followed by each one of the sites in which the processes are deployed. It also shows the steps followed by each site to deploy the processes. It is important to highlight that Site1 and Site 4 established a strategy to manage the resistance to change.

Table 1. General steps of the Deployment Process Methodology by sites

Activities	Site 1	Site 2	Site 3	Site 4	Site 5
1. Creation of the Quality and Process Committee.	X	X	X	X	X
2. Define and document the procedures with staff participation. Look for a model CMMI.	X			X	X
3. Identify the necessary changes. Modify and adapt the procedures.				X	
4. Elaboration of implantation plan	X			X	X
5. Perform a personal review of the procedures. Make the corrections.		X	X	X	
6. Communication of the implementation of new procedures.	X	X	X	X	X
7. Develop training strategy and conduct training as needed.	X	X	X	X	X
8. Procedures implementation.	X	X	X	X	X
9. The quality audits are planned.	X	X	X	X	X
10. Pilot the procedures. Make the corrections.	X	X	X	X	X
11. Evaluate the institutionalization degree.	X	X	X	X	X
12. Procedures implementation	X	X	X	X	X

3.3 Impact in Use of Processes

Each site implemented mechanisms for the use of processes. In order to validate the impact of an efficient change management when deploying new process in an organization, two indicators were considered: use of the deployed processes and level of acceptance by the staff. The value of these indicators was obtained from the answers to the survey that was filled in by the person in charge of each site. Specifically, the questions were:

- To indicate the status of the processes. The answer types were: full use, partial use, updating, and not used.
- To indicate the acceptance level of the processes by the project members. The answer types were: very high, high, medium, low and rejection.

Table 2 shows the obtained results of the survey related to the use of the deployed processes by site.

Table 2. Use of deployed processes by site

Processes Status	Site 1	Site 2	Site 3	Site 4	Site 5
Full Use	95%	30%	27%	78%	37%
Partial Use	5%	44%	54%	4%	18%
Updating	0%	13%	5%	0%	9%
Not Used	0%	13%	14%	18%	36%

At Site 1 and Site 4, the use of deployed processes percentage is greater than in the other sites. At Site 1, the staff that used the processes participated in their definitions. Site 4 tailored the processes from Site 5 in order to get the CMMI certification. Besides, Sites 1 and 4 had a deployment strategy to decrease the resistance to change. At Site 5, although they had defined their processes, they did not establish actions to reduce the resistance to change. As Sites 2 and 3 had not participated in the processes definition, their use was low.

Table 3 shows the results of the survey related to acceptance level of deployed processes.

Table 3. Acceptance level of deployed processes

Acceptance Level of deployed processes	Site 1	Site 2	Site 3	Site 4	Site 5
Very High	86%	4%	14%	10%	0%
High	9%	39%	43%	74%	38%
Medium	5%	22%	14%	16%	54%
Low	0%	35%	29%	0%	8%
Rejection	0%	0%	0%	0%	0%

As in the other question, in Site 1 and Site 4 the percentage of acceptance level (Very High and High) is greater than in the other sites. Sites 1 and 4 had an infrastructure supporting the processes (Quality Committee) dedicated full time to quality and process improvement tasks. In the other sites, the staff in charge of quality activities was also in charge of other activities.
Sites 1 and 4 established a strategy to manage the resistance to change including activities related to communication, motivation, training and marketing. The other sites did not have a defined strategy.
Sites 1 and 4 reached the CMMI Maturity Level 3 certification.

4 Conclusions

In this paper, the results from a process deployment work related to five software development and maintenance sites have been presented. The objective of this work was to analyze the performance of the processes deployment. In this case, the study is focused on the impact in the use and adoption of the deployed processes. Some aspects to highlight are the following:

- The survey shows there are great differences among the sites depending on the process deployment strategy. It is a key factor to design and plan a deployment strategy very early with dedicated resources. The plan is needed to get management support.
- It is necessary to have a Process Asset Library with easy access and ease of use by staff. The PAL allows knowledge sharing across the organization.
- Another key factor for successful deployment is to involve staff in the process definition.
- A strategy to manage resistance to change is fundamental for successful deployment. This strategy should consider aspects of communication, training and marketing. Do not forget that process deployment is focused on the human factor in order for the process to be adopted and used.
- The implementation of a model like CMMI in organizations is a change. It is necessary to consider the technical and human aspects.

Finally, in terms of future work, it would be appropriate to study the successful critical factors and their impact on process deployment.

Acknowledgments. This paper is sponsored by ENDESA, everis Foundation and Sun Microsystems through "Research Group of Software Process Improvement for Spain and Latin America". The authors would like to thank the participants from the case sites.

References

1. Bayona O., Sussy, Calvo Manzano, J., Cuevas, G., San Feliu,T. "Despliegue eficiente de procesos definidos y medidos (PAL) en múltiples organizaciones", (2007), http://lucio.ls.fi.upm.es/doctorado/Trabajos20062007/Bayona.pdf, Accesed 04-05-2008.
2. Carnegie Mellon, Software Engineering Institute. CMMI® for Development Version 1.2, CMU/SEI-2006-TR-008 pp 219-275 (2006)
3. Constantine, L. Peopleware Papers: The Notes on the human side of software, Prentice Hall (2001)
4. Daniel, D. Ronald. Management Information Crisis, Harvard Business Review, Sept.-Oct (1961)
5. DeMarco, T. and T. Lister, "Peopleware: Productive Projects and Teams", 2nd ed, Dorset House Publishing, New York (1999)
6. Garcia Suzanne, "What is a process Asset Library? Why should you Care": Aimware Professional Services. http://www.saspin.org/Saspin_Sep01_Boyd_Bandor.pdf, Accessed 04 January 2008
7. Kaplan, and N.T. Shaw, People, organization, and social issues: evaluation as an exemplar, in: R. Haux, C. Kulikowski (Eds.), Yearbook of Medical Informatics, Schattauer, Stuttgart/New York (2002)
8. Kotter Jhon P.Leading Change, Harvard Business School Press (1996)
9. Kulpa Margaret K. & Johnson Kent A. Interpreting the CMMI, A process Improvement Approach. Auerbach Publications (2003)
10. Mads Christiansen and Jorn Johansen, ImprovAbilityTM Guidelines for low maturity organisations, EuroSPI 2007 pp 91-105 (2007)
11. N.M. Lorenzi. R.T. Riley, Managing change: an overview, J. Am. Med. Inform. Assoc. 7 116-124 (2000)
12. Piattini Mario G., & Garzas P. Javier. Fabricas de software: Experiencias, tecnologías y organizacion. Madrid: RaMa Editorial (2007) (http://www.ra-ma.es/libros/0001915.htm).
13. Pries-Heje Jan and Malene M Krohn.Organizing Improvement Work; A longitudinal Case. LNCS, vol 4764, pp 91-105. Springer, Heidelberg (2007)
14. Software Engineering Institute.Carnegie Mellon University. CMMI Process Maturity Profile CMMI, Appraisal results 2006 End-year Update (2007)
15. Software Guru: Conocimiento en práctica. (2006) DOI=http://www.softwareguru.com.mx/downloads/SG-200606.pdf pp. 10-13
16. The International Process Research Consortium. A Process Research Framework, Eileen Forrester, editor. Software Engineering Institute (2006). (http://www.sei.cmu.edu/iprc/)

Author CVs

Sussy Bayona

She is a PhD candidate in Software Engineering at Polytechnic University of Madrid (UPM), graduated in Systems Engineer (1986) at the National University Engineering (Perú). She also received an MS in Software Engineering from the UPM (2004). Her research interest includes Software Engineering and Software Process Improvement focusing in Process Deployment, project management and software metrics. She has 16 years of experience as software engineer and consultant in public and private organizations. She is collaborating with a "Research Group of Software Process Improvement for Spain and Latin America".

Jose Antonio Calvo-Manzano

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.

Gonzalo Cuevas

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.

Tomás San Feliu

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.

Approaching Software Process Improvement to Organizations through the Reuse of their Processes and Products

Fuensanta Medina-Domínguez, Maria-Isabel Sanchez-Segura, Arturo Mora-Soto, Javier García Guzman, Antonio Amescua

Computer Science Department, Carlos III Technical University of Madrid
Avda. Universidad, 30, Leganes 28911, Madrid, SPAIN
{fmedina, misanche, jmora, amescua}@inf.uc3m.es

Abstract. It is widely known that the use of software engineering best practices in software projects development improves productivity and allows organizations to be more competitive. But sometimes the results obtained while using these best practices, as well as the way these best practices have been used are what allow the organizations to innovate their development processes, methods, etc.. This is part of what we call organization know-how and it has a high value for companies. In order to allow software organizations to reuse their know-how, the authors have defined product patterns artifact. This know-how can be used in combination with software engineering best practices to improve the quality and productivity of their software projects as well as to reduce projects cost. This paper describes the framework developed to encapsulate the know-how in organizations. The PIBOK-PB (Process improvement based on knowledge - pattern based) tool uses the knowledge encapsulated in the product patterns to execute software projects more efficiently. This paper also describes part of the PIBOK-PB tool development and compares similar tools in the market.

Keywords: Software Process Improvement, Process Assets, Product Pattern, Patterns, Knowledge Management, Reuse, Software Engineering.

1 Introduction

In spite of the fact that the use of software process best practices in a software project development improves productivity in organizations, projects planning and the quality of the software products [1], very few organizations implement these best practices [2, 3]. Currently, software process best practices are not deployed because of the little usability of project management tools. They present several constraints to represent, manage and transfer the knowledge of these best practices and the knowledge obtained through the software engineering experts' experience.

Knowledge management is defined as "the set of systematic and disciplined actions that an organization can take to obtain the greatest value from the knowledge available to it" [4]. Knowledge management is based on four elements [5]: *data* is the minimum unit of information not elaborated; *information* is a set of inter-related data to acquire sense; *knowledge* is defined [6] as expertise, and skills acquired by a person through experience or education; *innovation* is the successful exploitation of new

ideas. There are two kinds of knowledge: explicit and tacit. Explicit knowledge can be codified into documents and databases and shared among stakeholders relatively easily. Tacit knowledge is the know-how of organizations, the capabilities and experiences of their stakeholders. This knowledge is difficult to formalize, communicate and share among the people involved in the project in the organization. This tacit knowledge is essential to achieve the innovation stage so that the organization will gain the competitive advantage.

Software engineers look for knowledge sources, for example books, on-line resources, databases, to develop a software process but, on many occasions, this knowledge is the tacit knowledge of the organizations [7]. The study carried out by [8] revealed that the knowledge is not easy to find and, when it is discovered, it cannot be reused. The reasons are that knowledge gathering is too informal and the knowledge is not readily available to the organizations. Consequently, it is necessary to gather the know-how of the organizations in an artefact to be retrieved and reused in subsequent projects. We defined an artefact, called product patterns [9, 10], to gather the knowledge of the software engineering experts to produce a software product.

In this paper we describe the collaborative framework that supports software process improvement based on process assets reuse. The main capabilities of this framework are:

- to reuse and manage of process assets
- to improve the efficiency of use of the processes
- to reduce costs in software process improvement programs
- to work collaboratively in the phases of software projects

The remainder of this paper is structured as follows: section 2 describes the related works. Section 3 explains the collaboration framework proposed for process assets reuse in the project under development. And, finally, in section 4 we present the conclusions and future works.

2 Related Works

2.1 Patterns in Software Engnieering

In the mid 1960s, the architect Christopher Alexander came up with the idea of Patterns, as “A solution to a problem within a defined context” [11]. The pattern concept has clear origins, and an important value as a reusing tool. The main problem is that the word pattern has been used almost for everything, thus losing its original meaning.

Nowadays, in the field of software engineering there are many kinds of patterns. The authors have classified these patterns [12] as follow:

- Implementation patterns
    1. Reference architectural patterns
    2. Architectural patterns
    3. Analysis patterns
    4. Design patterns
- Process and improvement patterns

5. Process patterns
6. Software process improvement patterns

- Configuration management patterns
    7. Software configuration management patterns.

The conclusions extracted from the existing software patterns [12] reveal their low efficiency of use. Some causes are: most part of the patterns was described in natural language and are not software supported so they are difficult to reach, it is difficult to reuse patterns and their feedback execution are lose.
In some related works [13][14] the process patterns are defined to support development processes as a general solution to a certain recurring problem, but in the software development process, projects execute processes that produce software products, this is why we believe that the general solutions to be reused, are not the processes but the software products, and we propose the concept *"product pattern"* to gather the knowledge of software engineering experts to obtain a specific software product. More details about product patterns can be found in [9][10].These product patterns are described formally. Also, they are stored in a Process Assets Library and supported by a collaborative framework (see Section 3).

2.2 Existing Tools

The software tools analysed are presented, mainly focusing on the following features: knowledge management, know-how reuse, project management, software process models deployment support, collaborative development platform and knowledge searching.

- CodeBeamer [15]: is a collaborative development platform based on J2EE that offers application life cycle management features for development teams. It includes the following main capabilities: knowledge management (through a mechanism called wiki asset linking), project management and information retrieval. This tool lacks knowledge management formalism because wikis are considered content management systems [16] instead of a formal knowledge management artefact. This explains why knowledge reuse cannot be done automatically. Despite the fact that this tool is intended to support software development, it does not provide either a mechanism to select a software process model for the organization and project features or an electronic process guide to perform project activities.
- IRIS Process Author [17]: is a visual process management system that enables collaborative authoring and tailoring of process assets. This tool focuses on process management and, for project management, offers the possibility of exporting process content and configuration to third party tools such as Microsoft Project. Although this application also includes some knowledge management features, it lacks knowledge reuse for the organization and projects context and constraints. For know-how representation it uses templates and wikis instead of a formal representation artefact such as patterns, ontologies or a thesaurus. Another deficiency is the lack of software process models management, project planning and project tracking.

- Microsoft Visual Studio Team System (VSTS) [18]: is an integrated Application Life-Cycle Management (ALM) solution comprising tools, processes, and guidance to help the members of a development team to improve their skills and work more and more efficiently together. As mentioned in [10], VSTS presents some deficiencies. Therefore, we decided to develop a collaborative framework to cover the deficiencies related to knowledge management, knowledge reuse and project management.
- Select Solution Factory [19]: this tool connects component based development and solution assembly. It allows organizations to manage complex software development projects as well as to implement code and software components reuse. Although this application is intended to ease software development and deployment, knowledge management features are absent in reuse. This tool offers support for some modelling methods such as SSADM and Yourdon but lacks software process models management support.

Some prototypes were analysed according to the following features: knowledge management, know-how reuse, project management, software process models deployment support, collaborative development platform and knowledge tracking.

- BORE [20] is a prototype tool designed to further explore and refine the requirements for tools supporting experience-based approaches. This tool is aimed at reusing organizational experience by packaging it in experience repositories. Although this tool offers a huge functionality for knowledge management, its main flaw is project management. In spite of presenting tasks definitions for team members roles, it lacks vision of the project as a whole.
- OnSSPKR [21]: is a knowledge management based tool for supporting software process improvement. It is aimed at composing and organizing useful process assets into a knowledge repository based on ontology. It also supports software process knowledge storage for retrieval. This tool offers neither project management execution functionalities nor software process models management. It does not support a clear collaborative framework in spite having of web portals implemented.
- ProKnowHow [22]: is a knowledge management based tool for supporting software process improvement. It contains formal and informal knowledge in the software process database of an organization. It uses two ontologies to ground the structure of the organization's memory. This tools offers project management and some search capabilities for accessing past process plans and lessons learned. It lacks practical representation for knowledge reuse. Neither an active electronic software process guide nor a collaborative environment for project execution and management is offered.

3 Collaborative Framework Definition

PIBOK-PB is a collaborative framework to support software process improvement and is based on process assets reuse. With this tool the authors tried to fill all the gaps identified in the tools selected and explained in Section 2.

The PIBOK-PB architecture is shown in Fig. 1. The following sections describe the PIBOK-PB framework in more detail.

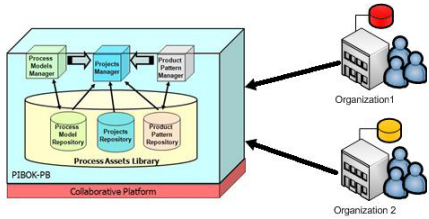


Fig. 1: PIBOK-PB Architecture

3.1 PIBOK-PB Framework

The main functionalities of the PIBOK-PB tool were described in [10]. The most relevant are: 1) the reuse of process assets supported by product patterns artefacts [9, 10]; 2) knowledge management capabilities; and 3) software process models selection for the organization and project features.

The collaboration layer was improved in the PIBOK-PB tool by developing a collaborative framework to fill the gaps in knowledge management, knowledge reuse and project management. The authors believe that this collaborative framework will facilitate the use of PIBOK-PB in organizations, and collaboration among those organizations that decide to share knowledge and information.

3.2 Knowledge Managements Components

The collaborative framework is made up of three knowledge management components as follows.

Software Process Model Manager: This component provides several software process models for each project under development. It offers a set of methodologies or activities to construct different processes such as software analysis, software design, software management, software development and software deployment processes. The project manager decides which model best fits the information provided by this component for the organization and project features. In addition, he/she can create a specific model process. In this way the organizations can customize their own software processes, creating a specific software process model.

Product Pattern Manager: This component provides the knowledge that a stakeholder needs to perform each process model activity. In order to achieve this task, the system looks for the product pattern that best fits each activity. In this search, the component takes into account the context, project and constraints of the organization, and the problem to be solved. The rule used to select the accurate pattern is:

```
If you find yourself in this context
   (and) with this problem
       (and) entailing these forces
    then
   map a product pattern in your project
   (and) look for more product patterns
```

All product patterns that best fit the input variables are shown to the project manager. One of the most important fields of product patterns is the solution. This field provides the steps to carry out this activity and obtain a specific software product. The project manager has to select and instantiate a product pattern for each activity based on his/her experience.

Current Project Manager: This component provides the execution plan for the current software project through an active electronic process guide and an electronic execution project guide. The active electronic process guide provides an overview of the software process model with the following elements: a software project planning, the tasks of each team member role and a workflow representing the processes of the project. The electronic execution project guide provides: 1) the product patterns showing the steps to obtain the defined software product; 2) templates where stakeholders can capture the information on the current project; 3) lessons learned by experts engineers; and 4) relevant information on each activity.

3.3 PIBOK-PB Process Assets Library Description

A process asset has no value if it is not easily accessible. The process assets have to be organized, well-indexed and easily accessible to stakeholders who need process guidance information. These characteristics are provided by a Process Assets Library (PAL).

The PIBOK-PB Process Assets Library is a collaborative knowledge base and a central repository that is made up of three repositories:

Process Model Repository: this repository stores the software process best practices, including software process models[1], methodologies[2] and techniques applied to software processes. The knowledge stored contains the activities and tasks of these software process best practices, roles of each activity and relevant information of each., This repository also contains a set of heuristics based on rules which, for a set of features of an organization and a set of project features to be developed, provides the process model, methodology and techniques that best fit these features.

[1] Capability Maturity Model Integration –CMMI- [23], ISO/IEC 15504, also known as SPICE - Software Process Improvement and Capability dEtermination- [24], etc.

[2] Unified Software Development Process –UP- [25], extreme Programming –XP [26], Personal Software Process –PSP [27], Team Software Process –TSP- [28, 29], Scrum Development Process, etc.

Product Patterns Repository: this repository stores the knowledge of software engineering experts in order to obtain a specific software product and the know-how of the organization. All this knowledge provides stakeholders with the information needed to create, develop and deploy new and better processes so that the organization achieves the innovation stage of knowledge management

Project Repository: this repository stores software projects that have already been developed as well as those under development. It contains three types of information related to:

- the process model chosen for this project: process model, activities, roles, workflow;
- the product patterns of each process model activity which provide enough information to perform each activity, and the active electronic process guide where this information is shown to the stakeholders;
- the execution and development of each activity where the information stored is: project planning, products obtained from each activity and project tracking.

3.4 PIBOK-PB Collaborative Support

The organizations need to share knowledge among stakeholders to enable them to develop the project activities, taking advantage of the knowledge of past projects and the experience and knowledge of the software engineering experts. In this way, stakeholders can perform their daily tasks and make decisions using the maximum information available. So, organizations have to commit themselves to providing the mechanisms and technologies to allow collaboration among stakeholders, coordinating activities and sharing knowledge to develop software project processes without increasing the time and cost of projects. In order to disseminate this knowledge, the framework is supported by a collaborative platform.

The collaborative platform proposed by the authors is based on the Microsoft Office SharePoint Portal Server (MOSS) because of the following capabilities:

- Better communication among different software projects roles.
- Coordination of the activities among teamwork members who need synchronous and/or asynchronous interaction regardless of their geographical location.
- Integration among different systems in the organization.
- Extension mechanisms through application programming interfaces.
- The most advanced search engine with many superior characteristics over its competitors such as meta data management or automatic language detection.
- A familiar and user-friendly interface.

3.4.1 PIBOK-PB Architecture Deployment

This solution is intended to be deployed in two different scenarios:

- If an organization wants to deploy a simple collaborative environment and does not have enough budgets to set up a complex collaborative tool, Microsoft Windows SharePoint Services 3.0 (WSS 3.0) can be installed because it is a free tool for those organizations or users using Microsoft Windows Server 2003 operative system. Data repositories of organizations using this configuration schema will be stored at Software Engineering Lab (SEL-UC3M) servers, observing data protection laws in order to respect the organizations' data. This must be done this way because WSS 3.0 does not include the Microsoft Enterprise Search, a very important component used to recover and reuse organizations know-how and knowledge. The proposed architecture for this scenario is shown in Fig. 2.

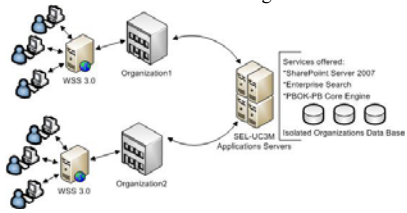


Fig. 2: Simple Collaborative Environment

- On the other hand, there is no problem if an organization wants to deploy a complete collaborative environment and also budget for licensing. Microsoft Office SharePoint Server 2007 (MOSS) needs to be installed. In this case, the organization's repository will be installed on-site giving full control and responsibility for its own data to the organization. This option also includes the Microsoft Enterprise Search and no organization's data will be installed in SEL-UC3M servers. The resulting architecture is given in Fig. 3.
- The PBOK-PB architecture is designed to allow organizations to share data among them as well as to attach existing data bases of any kind of system able to communicate using XML messages.
- The integration of new features focused in every organizations needs is also possible thanks to the extension capability of the collaborative platform used.

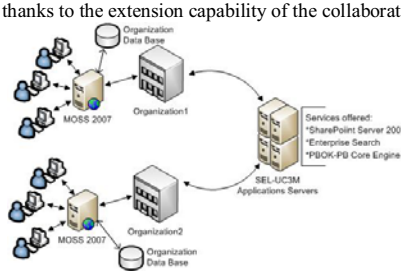


Fig. 3: Complete Collaborative Environment

3.4.1 PIBOK-PB tool development

The tool that supports the PIBOK-PB architecture presented above was analysed and designed using UML and object oriented techniques. The tool is now under development, so at the moment we have an evolutionary prototype.

Fig. 4 shows the diagram that summarizes the PIBOK-PB tool use cases, and

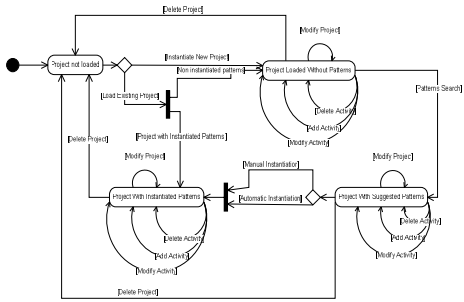


Fig.5 summarizes the main states which can be the part of the system corresponding to the component "product patterns manager".

In http://sel.inf.uc3m.es/documents/pbokpb_spi.pdf, we also describe the behavior of the system through the corresponding sequence diagrams. We have not included the above mentioned diagrams due to the extension of the paper.

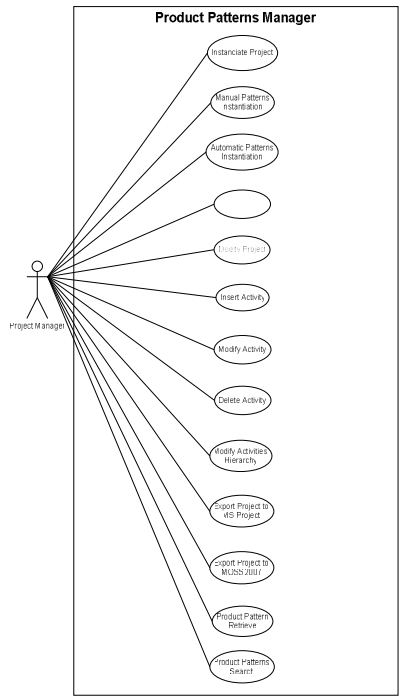


Fig. 4: PIBOK-PB tool use cases

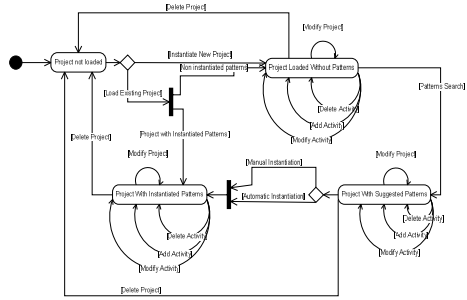


Fig. 5: Product Patterns Manager state diagram

The following is a screenshot of the collaborative framework prototype developed. This prototype is available at http://sel.inf.uc3m.es/demos/synergy/:

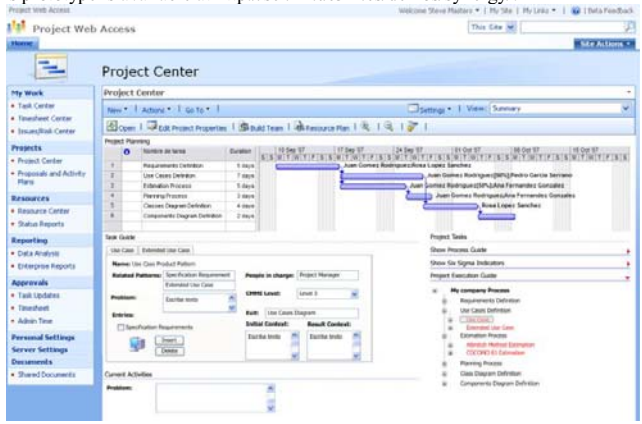


Fig. 6: Project Management and Tracking Interface

4 Conclusions and Future Trends

In this paper, we have described a collaborative framework to support software process improvement based on the reuse of process assets. The framework, called PIBOK-PB, is made up of tree knowledge management components, a process assets

library and a collaborative platform. Its main goal is the transfer and reuse of knowledge among the stakeholders involved in a software project development. The knowledge stored in the PAL contains the experience of software engineering experts, the know-how of the organization and the products obtained during the execution of any software project, usually stored as non structured assets.

The key benefits and potential use of our work include:

- Selection and customization of the software process best practices that best fit the needs of the organization's features.
- Reuse of the know-how of the organization and the knowledge of the software engineering experts through product patterns artefacts.
- Enough information to know, at each stage of the software process development, what each stakeholder has to do and the steps to take through an Active Electronic Process Guide.
- Knowledge sharing and working collaboratively through the collaborative platform.
- Feedback mechanisms so that new product patterns can be created and, consequently, promote innovation.

Acknowledgements. This work has been partially funded by the Spanish Ministry of Science and Technology through the TIC2004-7083 project.

References

1. Withers, D. Software engineering best practices applied to the modelling process. 432-439 (2000)
2. Richard, T. Seven Pitfalls to Avoid in the Hunt for Best Practices. 67-69 (2003)
3. Soumitra Dutta, M. L. Software Engineering in Europe: A study of Best Practices (1999)
4. Marwick, A.D. Knowledge management technology. IBM System Journal. Vol 4, pp. 40 (2001)
5. Hey, J. The Data, Information, Knowledge, Wisdom Chain: The Metaphorical link (2004)
6. Wikipedia. Retrieved January 3, 2008, from http://en.wikipedia.org/wiki/Knowledge
7. Assimakopoulos,D., Yan J. Sources of knowledge acquisition for Chinese software engineers. R&D Management (2006)
8. Komi-Sirviö, S., Mäntyniemi, A., Seppänen, V. Knowledge Management - Toward a Practical Solution for Capturing Knowledge for Software Projects. IEEE Software; 19 (3) pp.60-62 ISSN:07407459 (2002)
9. Amescua, A., Garcia J., Segura-Sanchez, M., Medina-Dominguez F. A pattern-Based Solution to Bridge the gap between theory and practice in Using process models. In LNCS, vol. 3966, pp. 97-104. Springer (2006)
10. Medina-Domínguez, F. Sanchez-Segura, M., Amescua, A, Garcia, J. Extending Microsoft Team Foundation Server Architecture to support Collaborative Product Patterns. LNCS 1-11 (2007)
11. Alexander, C. (1964). Notes on the Synthesis of Form. Cambridge, MA.: Harvard University Press

12. Medina-Domínguez, F. Sanchez-Segura, M., Amescua, A., Mora-Soto, A., Garcia,J. Patterns in the field of software engineering. To be published at *Encyclopedia of Information Science and Technology, Second Edition. Idea Group.*
13. Iida, H. Pattern-Oriented Approach to Software Process Evolution. (1999).
14. Hagen, M,, Gruhn, V. Process Patterns - a Means to Describe Processes in a Flexible Way. (2004).
15. Intland Software. codeBeamer. http://www.intland.com/products/codebeamer.html
16. Cunningham, W., Leuf, B. The Wiki Way. Addison-Wesley Professional (2001)
17. Osellus. IRIS Process Author. http://www.osellus.com/IRIS-PA
18. Microsoft Corporation. Visual Studio Team System. http://msdn2.microsoft.com/en-us/teamsystem/default.aspx
19. Select Business Solutions. Select Solution Factory. http://www.selectbs.com/products/select-solution-factory.htm
20. Henninger, S. Tool Support for Experience-Based Software Development Methodologies. Advances in Computers , 59, 29-82 (2003)
21. He, J., Yan, H., Liu, C., Maozhong, J. A Framework of Ontology-Supported Knowledge Representation in Software Process (2007)
22. Silveira Borges, L. d., Almeida Falbo, R. Managing Software Process Knowledge (2002)
23. CMMI SM Product Suite. http://www.sei.cmu.edu/cmmi/products/products.htm
24. ISO/IEC 15504(1-5):2005. Standard for Information Technology-Software process assessment (2005)
25. Jacobson, I., Booch, G., Rumbaugh, J. The Unified development process. Addison-Wesley (1999)
26. Beck, K. eXtreme Programming Explained. Addison-Wesley (2002)
27. Humphrey, W.: PSP(sm). A Self-Improvement Process for Software Engineers. Addison-Wesley (2005)
28. Humphrey., W. TSP(SM)-Leading a Development Team. Addison Wesley (2005)
29. Humphrey, W. TSP(SM)-Coaching Development Teams. Addison Wesley (2006)

30.