Bürsner, Simone (Ed.) et al.

Research Report

16th International Working Conference on Requirements Engineering: Foundation for Software Quality. Proceedings of the Workshops CreaRE, PLREQ, RePriCo and RESC

ICB-Research Report, No. 40

Provided in Cooperation with:
University Duisburg-Essen, Institute for Computer Science and Business Information Systems (ICB)

Suggested Citation: Bürsner, Simone (Ed.) et al. (2010) : 16th International Working Conference on Requirements Engineering: Foundation for Software Quality. Proceedings of the Workshops CreaRE, PLREQ, RePriCo and RESC, ICB-Research Report, No. 40, Universität Duisburg-Essen, Institut für Informatik und Wirtschaftsinformatik (ICB), Essen

This Version is available at:
https://hdl.handle.net/10419/58169

Simone Bürsner, Jörg Dörr,
Andreas Gehlert, Andrea Herrmann,
Georg Herzwurm, Dirk Janzen,
Thorsten Merten, Wolfram Pietsch,
Klaus Schmid, Kurt Schneider,
Anil Kumar Thurimella (Eds.)

ICB
Institut für Informatik und Wirtschaftsinformatik

16th International Working Conference on Requirements Engineering: Foundation for Software Quality

ICB-RESEARCH REPORT

Proceedings of the Workshops CreaRE, PLREQ, RePriCo and RESC

UNIVERSITÄT DUISBURG ESSEN

ICB-Research Report No. 40

October 2010

Die Forschungsberichte des Instituts für Informatik und Wirtschaftsinformatik dienen der Darstellung vorläufiger Ergebnisse, die i. d. R. noch für spätere Veröffentlichungen überarbeitet werden. Die Autoren sind deshalb für kritische Hinweise dankbar.

The ICB Research Reports comprise preliminary results which will usually be revised for subsequent publications. Critical comments would be appreciated by the authors.

Proceedings Edited By:

Simone Bürsner
Jörg Dörr
Andreas Gehlert
Andrea Herrmann
Georg Herzwurm
Dirk Janzen
Thorsten Merten
Wolfram Pietsch
Klaus Schmid
Kurt Schneider
Anil Kumar Thurimella

Contact:

Institut für Informatik und
Wirtschaftsinformatik (ICB)
Universität Duisburg-Essen
Universitätsstr. 9
45141 Essen

Tel.: 0201-183-4041
Fax: 0201-183-4011
Email: icb@uni-duisburg-essen.de

ISSN 1860-2770 (Print)
ISSN 1866-5101 (Online)

Abstract

This ICB Research Report constitutes the proceedings of the following four workshops which were held on Tuesday, 29th June 2010 as part of the Requirements Engineering: Foundation for Software Quality (REFSQ) conference 2010 at the University of Duisburg-Essen.

- First Workshop on Creativity in Requirements Engineering (CreaRE)
- First International Workshop on Product Line Requirements Engineering and Quality (PLREQ)
- First Workshop on Requirements Prioritization for customer-oriented Software-Development (RePriCo)
- First Workshop on Requirements Engineering in Small Companies (RESC)

Table of Contents

1 Preface

Since 1994, when the first REFSQ took place, Requirements Engineering (RE) continued to be a dominant factor influencing the quality of software, systems and services. In the past, RESFQ has been organised in conjunction with other conferences, mainly the International Conference for Advanced Information Systems Engineering (CAiSE). The strong interest in RE and the highly interactive event format of REFSQ contributed to REFSQ maturing to one of the leading international forums to discuss RE and its relations to quality. In 2010, for the first time, REFSQ was organised as stand-alone working conference. This provided the opportunity to initiate a series of REFSQ workshops.

We invited researchers and practitioners to submit workshop proposals to further intensify the exchange of early results, ideas, experiences and to stimulate new and emerging issues in the RE field. The final workshop programme included four half-day workshops on topics as diverse as Creativity in Requirements Engineering organised by Jörg Dörr, Andrea Herrmann, Klaus Schmid and Kurt Schneider; Product Line Requirements Engineering and Quality organised by Anil Kumar Thurimella, Dirk Janzen and Klaus Schmid; Requirements Prioritization for customer-oriented Software-Development organised by Georg Herzwurm, Benedikt Krams, Wolfram Pietsch and Sixten Schockert; and Requirements Engineering in Small Companies organised by Simone Bürsner, Thorsten Merten, Barbara Paech and Jörg Dörr.

We thank all workshop organisers for their efforts in attracting workshop papers, organising the reviews, holding the workshops and compiling the workshop proceedings. We would like to extend our thanks to Philipp Schmidt for his support to finalise this proceedings volume and to Paluno – The Ruhr Institute for Software Technology for sponsoring this proceedings volume. Finally, we are in debt of the supportive team who organised the workshop website, the location, the registration office and the local support on the conference premises.

Andreas Gehlert and Ernst Sikora

REFSQ Workshop Chairs

2 Technical Programme

3 First Workshop on Creativity in Requirements Engineering (CreaRE)

Editors

Jörg Dörr
Fraunhofer Institut IESE, Joerg.Doerr@iese.fraunhofer.de

Andrea Herrmann
University Braunschweig, andrea.herrmann@tu-braunschweig.de

Klaus Schmid
University of Hildesheim, schmid@sse.uni-hildesheim.de

Kurt Schneider
Leibniz Universität Hannover, kurt.schneider@inf.uni-hannover.de

Technical Programme

CREARE 2010
1st Workshop on Creativity in Requirements Engineering

Joerg Doerr[1], Andrea Herrmann[2], Klaus Schmid[3], Kurt Schneider[4]

[1] Fraunhofer Institut IESE, Germany, Joerg.Doerr@iese.fraunhofer.de
[2] University Braunschweig, Germany, andrea.herrmann@tu-braunschweig.de
[3] University of Hildesheim, Germany, schmid@sse.uni-hildesheim.de
[4] Leibniz Universität Hannover, Germany, kurt.schneider@inf.uni-hannover.de

1 Technical Program

The CREARE workshop took place as a half-day workshop on the 29th June 2010 in Essen (Germany). The following two presentations were given, followed by discussions:

- Daniel Kerkow, Sebastian Adam, Norman Riegel, Özgür Ünalan: **"A Creativity Method for Business Information Systems"**
- Luisa Mich, Daniel M. Berry, and Alessio Alzetta: **"Individual and End-User Application of the EPMcreate Creativity Enhancement Technique to Website Requirements Elicitation"**

2 Introduction

Requirements Engineering not only demands a systematic approach for eliciting, operationalizing and documenting requirements and for solving their conflicts, but requirements engineering also is a creative activity. It demands the stakeholders to create visions of a future software system and to imagine all its implications. Creativity techniques can support this creative part of requirements engineering.

This workshop brought together requirements engineers from industry and researcher who are interested in discussing the role of creativity in requirements engineering, and how and which creativity techniques can be applied to requirements engineering. The workshop served as a forum for the interchange of experience and research results. It also aimed at raising awareness in the RE community for the importance of creativity and creativity techniques.

The two presentations raised discussions about questions like: How to guide a creativity workshop? How much fun is allowed in software engineering? How to

investigate creativity in a repeatable and measurable way? How to measure the quality – instead of quantity – of ideas created?

There certainly is potential for future research on creativity in requirements engineering.

We thank our program committee for their support:

Maya Daneva, University of Twente, The Netherlands
Joerg Doerr, Fraunhofer Institut IESE, Germany
Ralf Fahney, FAHNEY Anforderungsingenieurwesen GmbH, Germany
Andrea Herrmann, University Braunschweig, Germany
Anne Hoffmann, Siemens AG, Germany
Sara Jones, City University London, United Kingdom
Daniel Kerkow, Fraunhofer Institut IESE, Germany
Klaus Schmid, University of Hildesheim, Germany
Kurt Schneider, Leibniz Universität Hannover, Germany
Roel Wieringa, University of Twente, The Netherlands
Konstantinos Zachos, City University London, United Kingdom

For further information, please see: http://sites.google.com/site/creare2010/

A Creativity Method for Business Information Systems

Daniel Kerkow, Sebastian Adam, Norman Riegel, Oezguer Uenalan

Information and Interactive Systems Department
Fraunhofer IESE
Fraunhofer-Platz 1
67663 Kaiserslautern, Germany
{daniel.kerkow, sebastian.adam, norman.riegel, oezguer.uenalan}@iese.fraunhofer.de

Abstract. This paper presents procedural guidance for performing activities of creative problem-solving within systems engineering. We have applied this method in four case studies in different domains, slightly varying the techniques applied. This paper describes the experience gained and gives recommendations. Furthermore, the procedure integrates itself seamlessly into an overall systems development framework.

Keywords: Requirements Engineering; Creativity; Innovation; Process; Best Practices

1 Introduction

How to get from a problem to a solution is a very psychological thing, and thus, very hard to formalize. Sometimes, ideas for improvement are obvious and do not need any specific creativity. It, for instance, a business process includes three sequential steps, which could also be done in parallel, the idea for improvement is directly visible. However, when new technology is needed to improve the efficiency or effectiveness of, or the satisfaction with, business processes significantly, it might be helpful to apply creativity techniques to identify innovations. Especially for companies striving to offer new products and services, innovations that go beyond solving simple problems must be found however, innovation does not evolve automatically, and even systematic engineering approaches do not lead to innovation without applying techniques that foster the latter. For this reason, we recommend developing ideas for improvement using established creativity techniques, e.g., [1], also in the context of business process**es.**

The decision on whether to apply these techniques in an explicit creativity workshop depends on the desired degree of innovation or the challenge to find good ideas for solving existing problems. The higher the desired degree of innovation or the higher the challenge to find good improvements, the more useful a creativity workshop typically is. Hence, the main purpose of a creativity workshop is to find innovative ideas for future products that go beyond the state of the art.

In this paper, we will first of all introduce the cognitive principles of creative problem solving, as they are described in psychological models of thought. Then, in section 3, we present the procedure we apply for planning and performing a creativity workshop. Finally, in section 4, we present four case studies, the differences in the procedure, and the lessons learned in these applications.

2 Principles of creativity

As Schmid presented in his thorough analysis of creative problem solving [2], there are several principles for creativity. Applying these principles to atomic cognitive elements (e.g., an idea, a concept, or a product) leads to a variation of these elements. The degree of variation can be expressed via probabilities. Very unusual variations that lead to highly innovative ideas are of inherently low probabilistic. We call such large steps, in which the components of an idea are significantly changed, replaced, restructured and transformed into something completely new, a "transformation". Beyond this, rather common variations are very probable. These variations are called exploration or combination.

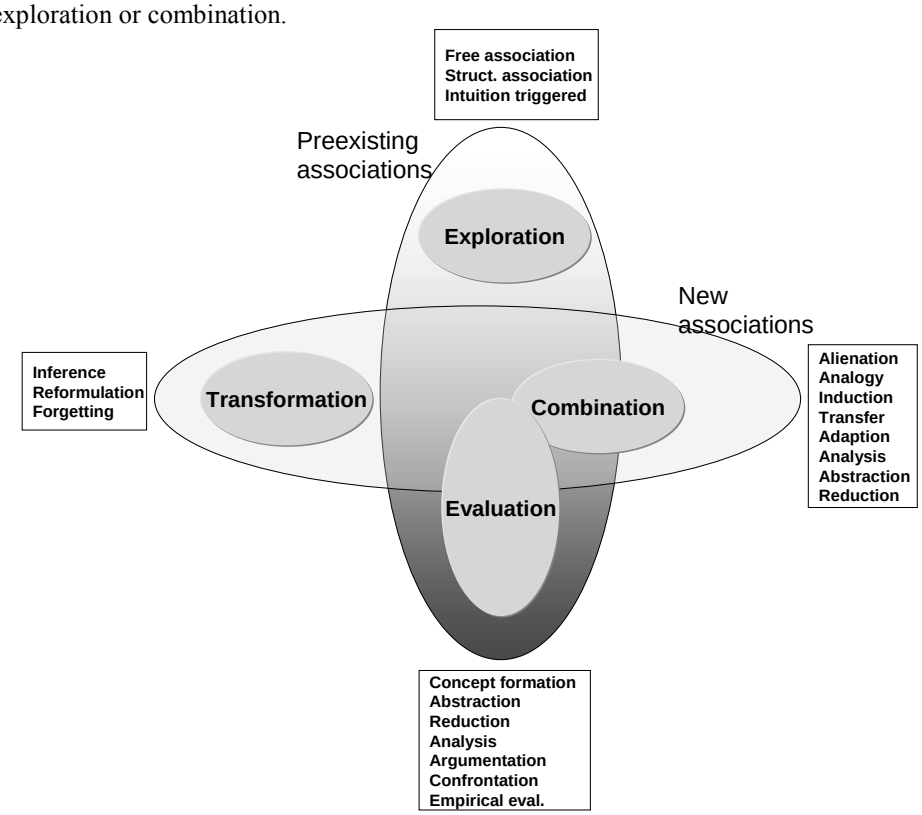


Fig. 1. Principles of creativity

Figure 1 illustrates some principles behind the different types of variation. The exploration of an idea space uses principles such as simple associations, for instance.

Combination, in contrast, rather requires alienation or analogy, whereas complete transformation uses principles such as inference or forgetting. Finally, evaluation is mainly based on argumentation or reduction. As Maiden et al. [3] show, creativity requires iterative oscillation from variation to evaluation, also known as divergence and convergence.

Besides these principles, typical problem-solving processes can be found in the literature, for instance CPS (creative problem solving) proposed by Boden [4] and IPC (information processing model of creativity) described by Schmid [2]. Both processes provide a procedural model of steps and mental activities that have to be followed in order to creatively solve a problem.

Based on these principles and processes, a multitude of creativity techniques have spung up. Prominent examples are Brainstorming [5], Morphological Analysis [6], Six Thinking Hats [7], Synectics [8], and TRIZ [9].

Our creativity workshop concept presented in the following section utilizes these techniques, among others, and integrates them into the process of information systems engineering. This process takes advantage of activities that according to CPS and IPC start with a preparation of preexisting information, such as problems, wishes ,and goals. It then continues with active and non-active problem-solving activities. Incubation is an important non-active step in creativity, in which the so-called "heureka" effect happens, i.e., uncontrolled restructuring steps in human cognition lead to sudden innovative findings. The importance of the incubation step has been empirically proven by Dodds et al. [10]. Finally the resulting ideas must be evaluated.

It has to be noted that our creativity workshop concept has been strongly inspired by the work of Maiden et al. [3], [11] and Schmid [2]. The novelty is therefore mainly the procedural integration into Fraunhofer IESE's requirements engineering approach "TORE" [12], as well as a procedural description of how to perform creativity workshops in the context of business information system development. Furthermore, while authors like El-Sharkawy et al. [13] focus more on semi-automated support for creative problem solving, our concept rather supports the human activity among participants of a workshop.

3 A Creativity Workshop for Business Information Systems

In this section, our creativity workshop concept for business information systems is introduced. As already mentioned, this concept is strongly influenced by the work of Maiden et al. [3], which we have evolved towards a standardized procedural method based on our own experience in recent years.

However, our standardized creativity method is not intended to be used at every level of abstraction. Rather, it is intended to support the transition from a given as-is (business) situation (e.g., traditional business processes, prevailing IT, or simply ways in which a user interacts with a technical system) towards an improved to-be situation (e.g., optimized business processes, including new technology, or new interaction metaphors). This means that the focus is on finding innovative solutions for problems that currently exist in business when an information system is used, while not going into too many technical details.

Our standardized creativity method consists of five phases and a parallel activity called "Storyboarding". Whether all phases have to be applied and which activities should be performed depends on the concrete purpose of the workshop. When someone likes to get a completely new and concrete concept, then all phases should be applied. Convergence in particular, is important in this case. When someone just wants to get innovative ideas or a vague vision, the exploration and combination phase are more important.

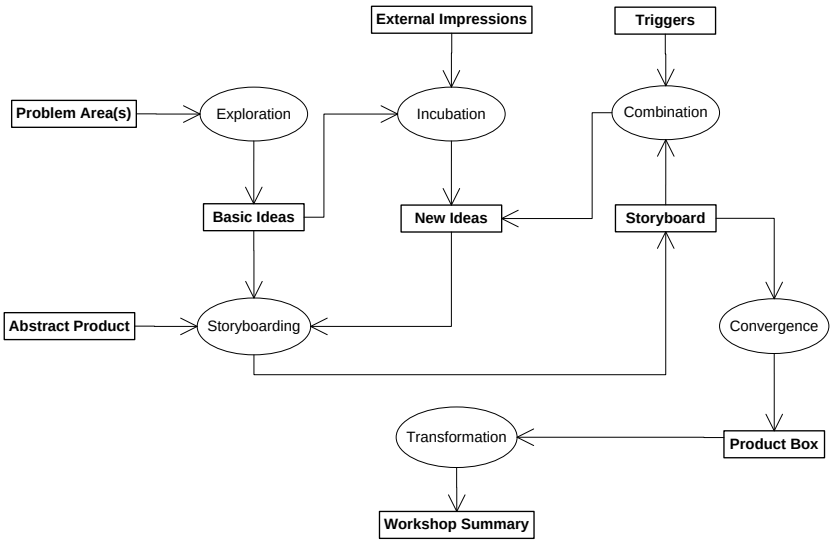


Fig. 2. Activities and objects of a creativity workshop

Figure 2 shows the activities (ovals) and objects (rectangles) handled during a creativity workshop. In the following, the five phases are described in more detail.

- **Exploration**: The purpose of the exploration phase is to get an initial set of ideas. This phase should always be the first phase in a creativity workshop and should enable the participants to bring in the ideas they already have on the topic of interest (typically a problem area elicited before). All associations the participants already have are then collected and written on small cards that are arranged on a table. After around 45 minutes, a set of eight main topics should be identified. Then, a second turn should be started and ideas the people have regarding each of these main topics should be collected again. A very important point to make at this stage is that all ideas are welcome and no judgment about their feasibility should be made at this point in time.
- **Storyboarding**: Storyboarding is not a phase but a parallel activity that should support the incubation, combination, convergence, and transformation phases. After collecting an initial set of ideas during the exploration phase, these ideas should be brought into a scenario describing how a business process should be performed with new IT support in order to solve the problem area from which it was started. Important rules are that as many ideas as possible from the exploration

phase should be integrated, and that the storyboard should be continuously adapted to new ideas found during the workshop.

- **Incubation**: The purpose of incubation is to let the mind meditate about the ideas identified so far, to develop new ideas, or to combine the ideas with concepts already read or seen somewhere else before. It is important to keep attention away from the topic of interest that so this process can be left to the subconscious mind. It is possible, for instance, to watch videos from similar domains, or just digress from the topic, e.g., by taking a walk. The incubation phase, in contrast to the subsequent combination phase, is a non-active phase, where the participants do not actively create or "do" anything. There is empirical evidence for the usefulness of this phase [10].
- **Combination**: The purpose of the combination phase is to investigate the ideas in greater depth and to develop new ideas that go beyond those already mentioned. Typically, three sub goals should be achieved. First, the existing storyboard and the existing ideas should be confronted with requirements a customer is interested in, or with constraints that are not negotiable (e.g., physical or political laws). This can be done by bringing in so-called trigger questions prepared for the workshop in advance. For instance, an important trigger question could be how the ideas developed so far are compliant with the security policies existing in the customer's organization. By bringing in such triggers, the participants are asked to think about it and to adapt their storyboard accordingly. A second concept within the combination phase is the analysis and variation of the storyboard's main properties and their values (e.g., using the method "morphologic box"). For instance, the work place could be identified as an important property in a scenario. Then, it should be analyzed whether the values currently assigned to the properties (e.g., "multitouch surface" as a value for the property "interaction device") in the scenario are the most appropriate ones or whether better values such as "mouse & keyboard" would be more appropriate. By analyzing alternative combinations, it is thus possible to explicitly assess the quality of each scenario property. Finally, when serious problems exist and the participants do not have any idea for solving them, a further technique ("force-fit-game") could be applied in which the participants are forced to be really creative by bridging words in a logical sense.
- **Convergence**: The purpose of the convergence phase is to find out which of the identified elements and ideas are really valuable for the intended purpose. Typically, many ideas are developed during a creativity workshop, but it seldom happens that all work or that all are related to the intended aims. Thus, defining a clear scope for the interactive system to be built is could be helpful, an important phase. To converge, the so-called "Product Box" technique [14], in which the stakeholders are asked to identify the really beneficial features found during the workshop day and to design a real product box that represents these features in a bold way. By building this box, the stakeholders develop arguments for why which feature could be helpful in a certain context. The Product Box technique should thus be applied as an effective prioritization technique. Especially the final presentation of these boxes helps to discuss in plenum the features and their benefit.
- **Transformation**: If the participants are not satisfied with the result of a convergence phase, it is possible to modify the defined "products" within the transformation phase. The transformation phase has the purpose of developing

variants of the "products" in order to assess whether they are more suitable than the original ones. This step results in a workshop summary describing the consolidated results.

However, a creativity workshop requires much preparation as it is very important to perform such a workshop in a relaxed, comfortable environment. Thus, an office or a normal meeting room is not appropriate, or needs to be prettified. Furthermore, even if a creativity workshop follows a clear method, it should be planned without any time pressure or strong agendas. All participants should be free to take breaks as they like. Only if the participants feel comfortable can they really be creative and develop innovations. For this reason, the techniques used in the workshop should be applied in a playful manner.

When participating in a creativity workshop for the first time, one might get a very esoteric impression. However, a creativity workshop is well founded in psychology, and works in the way that the applied techniques support people in their natural problem-solving capabilities. For instance, these techniques help to think about certain things explicitly, exchange existing ideas with other people, and combine or modify ideas with others.

Nevertheless, our creativity workshop concept does not deliver a solution that is directly implementable. Indeed, it delivers innovative ideas that have to be refined into clear requirements (e.g., to-be business process models, system use cases, etc.) , and it supports getting a shared and common vision on a certain aspect. This step again requires a certain amount of creativity but it can usually be done offline by experienced requirements engineers. We are currently preparing a method to formalize this step.

However, careful consideration of the results (especially with regard to cost effectiveness and technical feasibility) is still required in order to design a "real" interactive system. In this regard, it is important to document all results of such a creativity workshop, e.g., by creating visual minutes.

In table 1, the detailed procedure of how to prepare and perform a creativity workshop according to our method is proposed.

Table 1. Detailed procedure

Name	Develop ideas for improvement
Goal	Develop solution concepts that aim at improving the as-is processes.
Precondition	Problem areas within the as-is processes and constraints have been elicited.
Involved stakeholders	Representatives of the process stakeholders on the customer side and the developers (from the developer organization).
Procedure	**Prepare the workshop** 1. Analyze the number of required participants. Typically, you need at least four people per problem area to be discussed. If you would need more than 40 persons to address all problem areas, prioritize the problem areas

	based on your experience gathered in previous elicitation activities. 2. Organize a meeting and invite the participants, e.g., affected process representatives from the customer organization and representatives from the developer organization. If possible, talk to each person individually, sharing goals and expectations. If you address more than two problem areas, you will need two days for the workshop (due to the time required for the groups to present the results to each other). 3. Organize catering and a room for the workshop. 4. Prepare the room with whiteboards, flipcharts, cards, pens, pin board, cardboard boxes for each problem area, craft materials and tools, digital cameras, projectors, etc. 5. Arrange the room to create a comfortable atmosphere. 6. Consider the constraints identified in previous elicitation activities. If more than ten relevant constraints were identified, sort all soft constraints out. If still more than ten hard constraints exist, bring them into a prioritization order based on your experience gathered so far in the steps before. [Remark: Hard constraints are those which must hold in every case]. 7. Prepare cards 1. Write each problem area on a card. 2. If you already have in mind an abstract system you plan to develop, write the name of this abstract system (e.g., a new workflow system) on a card. 3. Transform each of the ten most important constraints into a trigger question or statement, and write each one on a card. 8. Collect any material you want to provide as external impressions (e.g., marketing videos, flysheets, etc.) **Perform the workshop [General rule: Take pictures of each end intermediate result]** 1. Welcome the participants and explain the purpose of this workshop. Give a short overview of what will be done without explaining all steps in detail. Encourage the participants to actively participate even if the workshop will probably be uncommon for them. 2. Divide the participants into groups to address the different problem areas. In each group place the related problem area card on the table, and explain the concrete meaning (e.g. by given examples from the as-is situation). 3. Explain the lotus blossom technique [15] to each group. Then, let each group develop an initial set of ideas. 4. After about one hour, encourage the groups to transform their basic ideas into a storyboard explaining how the problem will be solved by the ideas gathered so far. 5. When you get the impression that the groups are ready and have drawn their initial storyboards, take a joint (lunch) break. Use the break to give the participants the possibility to look at the external (impression) material. 6. After a one-hour break, let the groups come together again. 7. Read in plenum the triggers one by one and ask the groups to develop answers in their storyboards. Make the trigger cards available (e.g., on a pin board) so that everyone can think about them again. 8. When you think that each group has incorporated the triggers into its storyboard, explain to the groups the morphological box technique [16]. Then, let each group apply this technique to its current storyboard, i.e., the

	main properties of the storyboard and possible values for them are identified, and the "best" values are selected. 9. Ask the groups to modify their storyboards accordant to the results of the morphological box, if necessary. 10. If you have the impression that a group is not able to find appropriate solutions for the main problem area or an important sub-problem, apply the force-fit game [17]. 11. When each group has developed a final version of its storyboard, ask the groups to build product boxes [14] that converge the ideas they have developed into a packaged product solution. Typically, a product box should not contain more than ten high-level features. [If you have more than two groups, stop here on the first day]. 12. Ask the groups to present the product boxes to each other. Encourage them to point out the features and the benefits they see for product users. 13. Ask the other participants whether they consider this product appropriate for the intended purpose, i.e., for the problems actually existing in the as-is situation. If not, use SCAM(M)PER(R) techniques [18] to transform the product into a product that is considered more beneficial. 14. Write down all final features of all product boxes and related problem areas they address on a whiteboard. Allow each participant to assign up to 20 points to the features per problem area in order to prioritize the features. Ask the participants to assess the features according to the benefit they expect with regard to a solution of the actual problems they have. 15. Summarize the results of the workshop and give all participants the possibility to comment on the results as well as on the workshop itself. 16. Give a short outlook on how the results of the workshop will be used. **Consolidating the workshop results** 1. Consider all intermediate results (e.g., storyboards, initial idea cards gathered by lotus blossom technique, morphological boxes, product boxes, etc.) and check whether some interesting ideas are mentioned that were not integrated into the final feature list following on any of these SCAM(M)PER(R). Add these ideas to the feature list used for final prioritization. 2. Create a workshop summary including the extended feature list and feature descriptions in order to understand the ideas developed by the workshop participants.
Output	A list of ideas for improvement.

We are aware that there are many other creativity techniques that could also be applied (e.g., see [11], [14], [19], [20]), but we have made good experiences with applying the techniques proposed above. Be brave to experiment using different techniques that serve the purpose.

4 Application in four projects and synthesis

In this section, the characteristics and experiences gained in four applications of the creativity workshop in different projects are presented. These projects varied in terms of domain, number of participants, preparation efforts, and selection of creativity techniques. Table 2 gives an overview of the characteristics of each project, especially

the differences between them. Some characteristics proved to be beneficial for innovative results, while others showed to have a negative impact. Each column of table 2 represents a different project.

Table 2. Comparison and variations of the different workshops

	Project 1	Project 2	Project 3	Project 4
Domain	Public safety	Logistics	Forestry	Office Applications
Goal of WS	Identify innovative processes and tools	Identify innovative processes and tools	Identify innovative processes and tools	Identify innovative methodologies
Results	good	good	Abstract and not covering solution space (only problem space)	Rather visionary, not covering solution space. (only problem space)
Agenda	**Day1:** Divergence Convergence Lunch Incubation Divergence Dinner Incubation **Day2:** Convergence Evaluation	**Day1:** Divergence Lunch Convergence Evaluation	**Day1:** Divergence Lunch Convergence	**Day1:** Divergence Lunch Convergence
Ex-ploration method	Lotus Blossom	Lotus Blossom	Lotus Blossom	Lotus Blossom
Combi-nation/ Trans-formation method	Morphologic Box; Force Fit; Trigger	Futurescenario; Storyboard	Storyboard	Futurescenario; Storyboard
Incubation methods	Videos; Talks; Pictionary	Talks	None	Videos
Conver-gence method	Storyboarding; Product box	Storyboarding; Featurelist	Storyboarding;	Storyboarding; Product box
Evaluation method	Buy a feature	Give credits	Give credits	None
Degree of innovation	Medium	High	Low	Low
Number par-ticipants	40	29	20	12
Group rotation	None	Experts rotated among groups	None	None
Compo-sition of group	Well chosen according to interests and expertise	Well chosen according to interests and expertise	Ad hoc	Well chosen according to interests and expertise
Number of moderatos	4	3	2	2
Briefing of partici-pants	Well informed (written invitations and preparation workshop	Excellent (personal contact with each participant beforehand)	Low (goal and procedure were not communi-cated, participants	Excellent (personal contact with each participant beforehand)

	beforehand)		neither knew the project nor their role.)	
Environ-ment	Evening Event; Illumination; Music, Videos; Creativity workplaces	Creativity workplaces	None	Videos; Creativity workplaces
Input material	As-IS processes; Problems; Wishes; Use Cases	As-is processes	None	Roles; Goals; Exemplary Scenarios
Duration	1.5 days Including evening event	1 day	6 hours	1 day
Degrees of freedom	Medium (early in the project)	Low (late in the project)	High	Very high
Tools	Computer games; Gadgets; Trigger toys; Pinboard; Flipchart	Trigger cards; Pinboard; Flipchart	Trigger cards; Pinboard; Flipchart	Pinboard; Flipchart
Costs	High	Medium	Low	Low
Feedback	80% very satisfied 20% unhappy	Method: very satisfied Results: satisfied	Very unhappy (participants felt abused; results very abstract)	Satisfied (results: unhappy)

5 Synthesis of best practice

The most successful workshops were those performed in projects 1 and 2. The main differences between of these and projects 3 and 4 were the number of participants and the amount of time (and money) invested for preparation and performance. Creative processes need time and the benefit of different moods that change during the course of the day. We gained the best results with a duration of 1.5 days. Another very important factor is the briefing, selection, and compliance of the participants. In the following section, we summarize the most important practices:

Don't start too late in a project: Once a project reaches some significant progress, many thoughts are already there and many decisions have been inherently made. In late stages, it is very difficult to start thinking in new directions. Also, the benefits of the workshop seem rather low, since pre-existing ideas have mostly already found acceptance at that stage.

Plan enough time: Running out of time leads to unnecessary hurries and blocks creative thinking. Shortage of time also reduces the number of iterations between divergence and convergence. We experienced these iterations as beneficial for new ideas. We recommend spending 1.5 days for the workshop.

Have fun, but not too much: Creativity workshops are enjoyable; there is no doubt about this. There is a lot of communication, joking, talking about crazy ideas. In

between all this fun, one should not forget that the participants are working hard on these ideas, and that they probably have a personal commitment towards the project or the product. So remember to treat the results seriously. Communicate the benefits of the workshop and appreciate the efforts spent by the participants.

Don't oversimplify the prioritization and evaluation step: One of the most difficult and most important steps of the process is the prioritization step. This can be a hard task for the participants. The result of the workshop might be a list with 50 features on different levels of abstraction, and the participants are asked to choose three of them as the most important ones. Support this decision-making step and provide a method for multi-criteria evaluation.

Never start without prior analysis of the problem space in the domain: Our kind of creativity workshop needs something to start with. If you start from scratch, you will be able to talk about anything and lose the focus. So we strongly recommend performing domain, market, problem and requirements analyses beforehand. This will enable you to choose the right experts as well as the best gadgets, videos, and talks. With the right amount of preparation, focusing on specific problems and moderating the group areas becomes an easy task.

Don´t rotate participants unless there is enough time: It seems to be a good idea to exchange participants among groups at a certain point in time. We made the experience that this leads to a loss of time. All information shared before must be repeated. Of course, the interference of the new participants can be very useful for the evolution of the ideas, but it takes a lot of time. We only suggest rotating participants when there is plenty of time.

Contact people personally beforehand and clarify expectations and goals: Compliant participants are your most important factor for succeeding in creativity. So take the time to call every single person before the workshop. Ask for goals and expectations; explain the procedure and the necessary preparation. Take the participant on board. Otherwise the workshop will most probably fail.

Include good Incubation: The necessity of an incubation phase has been shown by Dodds et al. [10]. So be courageous and plan incubation. Don't be afraid of participant's reactions to things like taking a walk, watching videos, playing games, or just practicing an hour of yoga.

Choose participants carefully: Take care to ensure optimal composition of the group. Try to screen the participants beforehand. There are people who are open for creative thinking and supporting techniques, and there are others who reject these. If you have the chance to elicit these characteristics beforehand: Do so and don't invite the narrow minded fellows to the workshops. Include them in later or earlier steps, but don't let them slow down the creative group process. Also try to install groups with an interesting mix of personal interests. Note that this does not mean to generate homogeneous groups, the interests may also be heterogeneous. The important thing is to plan the mix carefully.

Use creativity techniques to enforce variations: When you start to propose creativity techniques to people beyond the classic brainstorming, you might get strange feedback. People might question the benefits and procedure of the techniques. They might say: "We are able to think without instructions". If this is the case, you might already have made a mistake, or chosen the wrong participants. However, try to motivate the techniques anyway and apply techniques that strongly transform the ideas and concepts during the workshop (such as the force-fit game). The most innovative ideas result from these techniques.

Prepare the convergence step carefully, plan breaks: Exploration activities (such as lotus blossom) create many ideas and thoughts. They might create so many ideas that the group gets completely lost and paralyzed. So make sure that convergence steps are well prepared. You should know beforehand how to reduce the ideas again, and how to restructure the concept. So prepare a "container" for the convergence, e.g., a worst-case scenario, which is improved during the workshop, applying the new ideas.

Atmosphere: In order to have unusual ideas, people should meet in unusual places. So if possible, invite to unusual locations with a nice atmosphere or at least create a good artificial atmosphere, changing illumination, playing music, etc. Think of the creativity workshop like a small party. Prepare drinks and food, create possibilities for individuality (different types of seats). The most important part of the atmosphere is the moderation. Moderators frame an open-minded atmosphere with the right choice of instructions. So make sure to invite skilled moderators to your workshop.

Fig. 3. Impressions of creativity workshops at Fraunhofer IESE

6 Conclusion

We have presented a systematic procedure for applying creativity techniques in requirements engineering processes for business information systems. We have seen that such a procedure cannot guarantee innovative results and that using the techniques proposed is not always the best choice. Nevertheless, our procedure gives the inexperienced innovation seeker a good starting point and hopefully prevents others from repeating some of our basic mistakes. Another shortcoming is the amount of effort and time needed by all participants to perform such a creativity workshop. This could be avoided by using web-based technologies (e.g., wikis, forums) in order to collaboratively generate ideas. Even if the creativity workshop concept presented above gives an impression of following a recipe in a cooking book, creativity remains a very dynamic and human thing, which highly depends on the preparation efforts but also on the personal skills and cognitive flexibility of the participants. We are looking forward to future work in formalizing creative problem solving and most of all in using these results in the process of innovative product development.

References

1. Clegg, B., Birch, P.: Instant Creativity: Simple Techniques to Ignite Innovation & Problem Solving, Kogan Page (2007)
2. Schmid, K.: Making AI systems more creative: the IPC-model, Knowledge-based systems, vol. 9, pp. 385-397 (1996)
3. Maiden, N., Robertson, S., Robertson, J.: Creative requirements: invention and its role in requirements engineering, Proceeding of the 28th international conference on Software engineering, Shanghai, China: May 20-28 (2006)
4. Boden, M.A.: The creative mind: myths & mechanisms (1991)
5. Osborn, A. F.: Applied imagination: Principles and procedures of creative problem solving (Third Revised Edition), Charles Scribner's Son, New York (1963)
6. Zwicky, F.: Discovery, Invention, Research - Through the Morphological Approach, The Macmillian Company, Toronto (1969)
7. de Bono, E.: Six Thinking Hats: An Essential Approach to Business Management. Little, Brown, & Company (1985)
8. Gordon, W. J.: Synectics: The Development of Creative Capacity, Harper and Row, New York (1961)
9. Altschuller, G., Seljuzki, A.: Flügel für Ikarus: Über die moderne Technik des Erfindens, Verlag MIR Moskau und Urania-Verlag, Moskau (1983)
10. Dodds, R.A., Smith, S.M., Ward, T.B.: The use of environmental clues during incubation. In: Creativity Research Journal, 14, 287-304 (2002)
11. Maiden, N., Manning, S., Robertson, S., Greenwood, J.: Integrating Creativity Workshops into Structured Requirements Processes. In: Proceedings of the 5th conference on Designing interactive systems, ACM (2004)
12. Paech, B., Kohler, K.: Task-Driven Requirements in Object-Oriented Development. In: Perspectives on Software Engineering, Kluwer Academic Publishers (2004)
13. El-Sharkawy, S., Grube, P. P., Schmid, K.: Using Semantically Linked Content to Support Creativity in Product Innovation: Computationworld, pp.638-642, 2009 Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns (2009)

14. Hohman, L.: Innovation Games: Creating Breakthrough Products Through Collaborative Play, Addison-Wesley Professional (2006)
15. Michalko, M.: Creative thinking technique: Lotus Blossom, http://www.innovationtools.com/Articles/ArticleDetails.asp?a=160, published (2004)
16. The Future Group: RELEVANCE TREE AND MORPHOLOGICAL ANALYSIS. www.futurovenezuela.org/_curso/12-tree.pdf, last visit 2008/04/29.
17. Mycoted since & technology: Force fit game. http://www.mycoted.com/Force-Fit_Game, last visit 2008/04/29
18. Mycoted since & technology: SCAMMPERR. http://www.mycoted.com/SCAMMPERR, last visit 2008/04/29
19. Bayerl, C.: 30 Minuten für Kreativitätstechniken, Gabal (2005)
20. Hargadon, A.: How Breakthroughs Happen: Harvard Business School Press (2005)

Individual and End-User Application of the EPMcreate Creativity Enhancement Technique to Website Requirements Elicitation

Luisa Mich[1], Daniel M. Berry[2], and Alessio Alzetta[1]

[1] Department of Computer and Management Sciences, University of Trento
I-38100 Trento, Italy
luisa.mich@unitn.it

[2] Cheriton School of Computer Science, University of Waterloo
Waterloo, ON, N2L 3G1 Canada
dberry@uwaterloo.ca

Abstract. This paper describes a preliminary experiment, involving individuals of two distinct kinds of stakeholders of a tourism Web site, whose results support a tentative conclusion that the EPMcreate creativity enhancement technique, normally used by groups of requirements analysts, can be used by individuals and by domain-expert end users.

1 Introduction

Many have observed the importance of creativity in requirements engineering, e.g., [1–3]. Many creativity enhancement techniques (CETs), e.g., brainstorming [4], have been developed to help people be more creative. Some of these CETs have been applied to requirements engineering [5, 2], and some have also been subjected to experimental validation of their effectiveness [5, 6]. A full discussion of these CETs can be found elsewhere [7].

This paper investigates the use of the creativity enhancement technique CET, *EPMcreate (EPM Creative Requirements Engineering [A] TEchnique)* [7, 8]. The feasibility of applying EPMcreate to idea generation in requirements elicitation was established by experiments on two computer-based system (CBS) development projects with very different characteristics. Each experiment compared the requirements idea generation of two analysis teams, one using EPMcreate and the other using brainstorming [7]. The results of these first experiments confirmed that, in at least the situations of the experiments, EPMcreate:

1. can be used by analysts, both junior and senior, requiring only minimal training and
2. produces more ideas and, in particular, more innovative ideas than does brainstorming.

Another investigation [8] compared with the 7Loci Metamodel the quality of the ideas produced by the two treatments in these same experiments and concluded that EPMcreate produced more ideas related to content and service requirements than did brainstorming.

The first experiments exposed a number of issues to be explored in the future. These include the questions:

1. Can an individual use EPMcreate as well as a group does?
2. Can a domain-expert, end user use EPMcreate as well as a system analyst does?

The purpose of the research reported in this paper is to begin to answer these two questions by an experiment that tests parts of these issues in the context of eliciting requirements for a promotional Web site.

In the rest of this paper, Section 2 describes the EPMcreate CET. Section 3 describes the preliminary experiment, including hypotheses. Section 4 gives the results of the experiment. Section 5 discusses whether the hypotheses are supported, and Section 6 concludes the paper.

2 The EPMcreate CET

A page limitation forces the description of the EPMcreate CET in this section to be brief, omitting all but what is necessary to understand the experiment and the results. However, EPMcreate is described fully elsewhere [7].

EPMcreate supports idea generation by focusing the analyst's search for ideas on only one logical combination of two stakeholders' *viewpoints* at a time. Sixteen such combinations are possible, each corresponding to one of the Boolean functions, fi for $0 \leq i \leq 15$, of two variables. The interpretation of some of these functions in terms of combining the viewpoints of stakeholders SH1 and SH2 are:

$f1$ = SH1 $\wedge$ SH2 represents ideas that SH1 wants and SH2 wants.
$f2$ = SH1 $\wedge \neg$SH2 represents ideas that SH1 wants but SH2 does not want.

To use EPMcreate to generate requirement ideas for a CBS, an analyst first identifies two of the CBS's stakeholders whose viewpoints she will explore by mentally playing their roles. In each of the sixteen steps of EPMcreate, she will explore a different combination of the four regions in the Venn diagram of Figure 1. In this diagram, the two

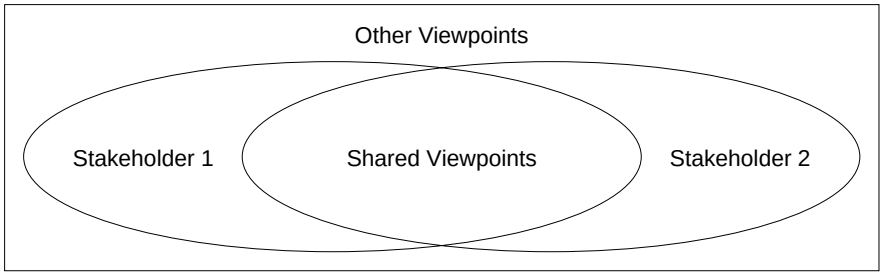


Fig. 1. Venn Diagram of Two Stakeholders' Viewpoints

ellipses represent two different stakeholders' viewpoints. Thus, for example, the intersection region represents the stakeholders' shared viewpoints, $f1 = \text{SH1} \wedge \text{SH2}$. In any step, the analyst uses the Boolean function that names the step to mentally combine the chosen stakeholders' viewpoints to trigger creative ideas.

If there are more than two types of stakeholders, EPMcreate can be applied several times, for each relevant pair of stakeholder types.

3 The Experiment

This paper describes an experiment that was designed to partially address the two questions raised in the introduction. The starting hypotheses were:

H1 The EPMcreate CET can be applied for requirements elicitation by an individual.
H2 The EPMcreate CET can be applied for requirements elicitation by a domain-expert end user.

Hypothesis H1 is important both because each CET is often classified as either an individual or a group technique and because the feasibility of EPMcreate as a group technique has already been demonstrated [7]. Being able to use EPMcreate as an individual technique would help reduce the costs of using EPMcreate to identify requirements. For example, brainstorming was defined as a group CET, but was shown to be applicable also by an individual [9, 5].

Hypothesis H2 is important because best practices in requirements engineering suggest end-user involvement in the requirements processes, including elicitation [1].

Note that these hypotheses fall short of fully addressing the two questions, because they do not *compare* individual use of EPMcreate to group use and domain-expert end user use to system analyst use.

3.1 Design of the Experiment

Designing an experiment to check the hypotheses, which are about the EPMcreate CET, required four main decisions, each covered by one subsubsection below:

3.1.1 Choosing the CBS to be Subjected to Requirements Elicitation In choosing the CBS whose requirement ideas were to be generated, we observed that nowadays many CBSs are Web based. Therefore, we decided to use a Web site as the CBS and chose the Web site of a jazz festival in the Dolomites, FiemmeSkiJazz, `http://www.fiemmeskijazz.com/`, which was renamed "DolomitiSkiJazz" in 2009. The festival's program includes jazz concerts and jam sessions offered by jazz musicians from the entire world, performing in ski lodges of the Dolomites.

3.1.2 Choosing Stakeholders as the Source of Viewpoints and Choosing their Representatives According to tourism marketing principles [10], the stakeholders for such a Web site can be classified into two main categories, *producers* and *consumers*[3]:

[3] The normal words are "organizers" and "users". We use "producers" and "consumers", respectively, to avoid confusing the latter with "end user", which has a different meaning in this paper.

1. Among the producers of the Web site and its festival are (1) the local tourist office, (2) the collection of friends that had the initial idea for the festival and that annually contacts artists for the concerts, (3) the Web-site analysts, (4) the sponsors, the partners, and (5) any tour operators that may be involved in organizing the festival.
2. Among the consumers of the Web site are (1) tourists, (2) musicians, and (3) occasional visitors of the Web site.

We chose Web-site analysts as the representative of the producers and musicians as the representative of the consumers. Certainly, Web-site analysts are the most technical of the producers and musicians are the most domain centered of the consumers. Nevertheless, that the chosen kind of producers and consumers are not representative of producers and consumers, respectively, is a possible threat to the validity of the experiment. We considered both producers and consumers as domain-expert end users for the purposes of Hypothesis H2.

3.1.3 Choosing the Subjects of the Experiment We identified many potential analyst subjects among the students who had successfully completed an Economics degree's undergraduate Web-site-engineering course, which focused on Web-site quality. While these potential subjects were not yet professional analysts, because of their course, they could be considered equivalent to junior analysts. They could even be considered more competent than senior analysts to evaluate Web sites because they had only recently completed the Web-site-engineering course. We identified 13 musicians we knew as potential musician subjects. None of these musicians were current or former students in Economics; so none had taken the Web-site-engineering course. As is typical among musicians, most could not survive on music alone; so many had so-called day jobs in various professions. All had some experience with computing, but these days, it is really hard to find anyone in their young age group without *some* computing experience. Therefore, these musicians were quite typical. We contacted these potential subjects by e-mail, asking them to participate in our experiment. Seven students and 6 of the 13 musicians agreed to participate in the experiment.

3.1.4 Evaluating Generated Ideas In the experiment, the effectiveness of the CET, EPMcreate, is measured by two numbers about the ideas generated when using the CET: (1) the *quantity*, i.e., the raw number, of ideas and (2) the *quality* of the ideas, as measured by the 7Loci Metamodel of Web-Site Quality.

The raw number of ideas generated is commonly used to measure the effectiveness of CETs [5, 7, 11] if for no other reasons than that the best known CET, brainstorming, encourages quantity over quality in its first step and that other CETs are compared to brainstorming [7].

Previous work by Mich, Berry, and Franch shows that the 7Loci-Metamodel assessment of ideas carried out by 7Loci-Metamodel experts who were not experts in the Web application's domain was essentially the same as the more subjective assessment of the same ideas carried out by experts in the Web application's domain [8]. Therefore, when objectivity is needed, as in conducting experiments, it is acceptable to use the 7Loci Metamodel to evaluate the quality of ideas.

Among the dimensions of the 7Loci Metamodel, *Identity* concerns the image that the organization projects and all elements that work together to identify the site's owner.

Content concerns the information available to the consumer, and *Services* concerns the services available to consumers. *Location* concerns the site's visibility and whether there is a place from which consumers can communicate with the organization and with each other. *Maintenance* concerns guaranteeing proper functioning and continued operation of the site, while *Usability* concerns how accessible and user friendly are the content and services of the site. *Feasibility* concerns initial and continued implementability and management of the site's services and project.

For the purpose of the later analysis, it is useful to classify each dimension into one of three groups: (1) *semantic*, (2) *syntactic*, and (3) *pragmatic*. Content and Services are semantic dimensions; Location, Maintenance, and Usability are syntactic dimensions; and Identity and Feasibility are pragmatic dimensions.

3.2 Realization of the Experiment

The experiment was carried out on 23 October 2007 in a computer room of the Faculty of Economics at the University of Trento. Each subject could visit the Web site as he or she pleased and could write his or her ideas for requirements in a Word file that initially contained only the description of each step of EPMcreate. That Word file led the subject through the steps he or she was to follow. The duration of the experiment was one hour, not including the five minutes spent giving to the subjects preliminary instructions for the procedure to be followed in applying EPMcreate to the problem at hand.

4 Analysis of the Results

To properly interpret the results of the experiment, it was necessary to be able to preclude that differences in the results were due to differences in the creativity of the subjects. For this purpose, as in earlier experiments [7], we used an adult version of Frank Williams's Creativity Assessment Packet [12]. Results of the testing confirmed that the creativity levels of the two kinds of subjects were almost the same: the analysts' average score was 69.71 out of 100 and the musicians' average score was a very close 71.67.

Each requirement idea was extracted from each subject's file and classified according to its possibly multiple dimensions. Duplicate requirement ideas from one subject were eliminated and thus not counted more than once. Any sentence containing more than one requirement idea was broken into atomic requirement ideas and each atomic idea was evaluated separately. The evaluation of requirement ideas was supervised by a senior, professional analyst, who was neither a subject nor an experimenter.

4.1 The Data

Table 1 summarizes the data yielded by the experiment. In this table,

- for each kind of subject and for each dimension,
  - the number in the column labeled by "# ideas" is the count of ideas generated by all subjects of the kind for the dimension,
  - the number in the column labeled by "%-age" is the percentage of the total count of ideas generated for the dimension that the number to its left is, and

  - the number in the row labeled "%-age" is the percentage of the total count of ideas generated by the kind of subject that the number above it is; and
- for each dimension, the number in the column labeled by "Total # ideas" is the count of ideas generated by all subjects for the dimension.

Because we had 7 analysts and 6 musicians, it was necessary to normalize the raw number of requirements ideas per dimension for any kind of subject into an average by dividing the raw number by the number of subjects of the kind. These raw numbers and their corresponding averages are shown in Table 2; this table shows also the value of the Student's T-test for each dimension. This table shows that each kind of subject generated a large number of requirement ideas, 174 by the analysts and 164 by the musicians. However, the average number of requirement ideas generated per subject of the two kinds are very similar, 24.86 by the analysts and 27.50 by the musicians. The same can be said for the average number of requirement ideas generated per subject for each dimension. The classification of the ideas was carried out by author Alzetta, and his classification was validated by author Mich.

The table shows by the absence of a row for "Feasibility" that none of the generated requirement ideas was classified as a feasibility requirement.

For each kind of subject, a plurality of its generated requirement ideas were classified into the Content dimension, a semantic dimension, consistent with the fact that the Web site is mainly informative. The other semantic dimension, Services, ranked third. All together, the semantic dimensions, Content and Services, that play a very important role for the success of a Web site, contain 49.26% of the requirement ideas generated using EPMcreate. The syntactic dimensions, Identification, Management, and Usability, contain 30.39% of the requirement ideas generated using EPMcreate; finally, the pragmatic dimension, Identity, contains 20.35% of the requirement ideas generated using EPMcreate.

The biggest differences between the analysts and the musicians are in the average numbers of Management and Usability requirement ideas their individuals generated. The average analyst generated 0.57 Management and 3.14 Usability requirement ideas, while the average musician generated 0.17 Management and 5.50 Usability requirement ideas. Note that the Student's T-test values for these differences are the highest and are nearly the same, at 1.1315 for Management and 1.1310 for Usability. These differences make sense when the expertises of the two kinds of subjects are considered.

- There were very few Management requirement ideas and analysts naturally found more of them than musicians.
- Knowing jazz music, appears to account for the musician's finding more Usability requirement ideas. A typical musician knows more than a typical analyst what could improve the user's navigation and experience in a musical event's Web site.

The average analyst generated 4.00 Services requirement ideas, while the average musician generated 5.17 Services requirement ideas, and the Student's T-test value for this difference is 0.6996. However, the average analyst generated 8.86 Content requirement ideas, while the average musician generated 7.67 Content requirement ideas, and the Student's T-test value for this difference is 0.4523. Thus, the Student's T-test gave a higher relevance to the first of these differences.

Table 1. Classifications and Numbers of Requirements Found by Analysts and Musicians

7Loci Dimension	Analysts		Musicians		Total
	# ideas	%-age	# ideas	%-age	# ideas
Identity	**37**	53.62	**32**	46.38	**69**
%-age	21.26		19.39		20.35
Content	**62**	57.41	**46**	42.59	**108**
%-age	35.63		27.88		31.86
Services	**28**	47.46	**31**	52.54	**59**
%-age	16.09		18.80		17.40
Identification	**21**	48.84	**22**	51.16	**43**
%-age	12.07		13.33		12.68
Management	**4**	80.00	**1**	20.00	**5**
%-age	2.30		0.60		1.49
Usability	**22**	40.00	**33**	60.00	**55**
%-age	12.65		20.00		16.22
Total	**174**		**165**		**339**

Table 2. Average Numbers of Requirement Ideas and Student's T-test Values

Dimension	Analysts		Musicians		Student's T-test
	Tot.	Avg. of 7	Tot.	Avg. of 6	
Identity	37	**5.29**	32	**5.33**	0.0254
Content	62	**8.86**	46	**7.67**	0.4523
Services	28	**4.00**	31	**5.17**	0.6996
Identification	21	**3.00**	22	**3.67**	0.4407
Management	4	**0.57**	1	**0.17**	1.1315
Usability	22	**3.14**	33	**5.50**	1.1310
Total	174	**24.86**	165	**27.50**	

The facts that

1. the average total numbers of ideas generated by the analysts and by the musicians for each dimension and
2. the numbers of ideas generated by the analysts and by the musicians for each dimension

were not statistically different could be interpreted, on one hand as the main characteristic of EPMcreate, that asking a subject to focus on different viewpoints allows also subjects not expert in the domain to better understand a variety of user needs.

That musicians suggested very specific services can be explained by their knowledge of what a Jazz musician needs. For example, one musician proposed creating an online auction to sell pairings of unknown Jazz performers with well-known Jazz performers.

It is interesting to observe that each kind of subject generated a high number of Identity requirement ideas, 5.29 by the average analyst and 5.33 by the average musician.

These numbers and the Usability numbers suggest that users can be involved not only for Usability requirements, but also Identity requirements. After all, image-related issues are very important for all stakeholders of a Web site and ultimately for the success of the Web site.

4.2 Evaluation of the Hypotheses

This section discusses how much the data support any of the hypotheses.

Hypothesis H1 is that the EPMcreate CET can be applied for requirements elicitation by an individual, instead of by a group. The results of Section 4.1 show that the average number of total requirement ideas per individual was 24.86 for analysts and 27.50 for musicians. The question is, "Are these satisfactory numbers?"

Comparing these data with those of the past experiments with groups [7] allows drawing a tentative conclusion that EPMcreate is effective when used by individuals. Each of these past experiments involved two groups of 4 subjects, one group applying brainstorming and one group applying EPMcreate, generating requirement ideas for one Web-based application. Each of these applications was larger than the Web site used for the current experiment, but the creativity sessions for them lasted 120 minutes and 100 minutes respectively, as opposed to 60 minutes for FiemmeSkiJazz. For one application, the EPMcreate group generated 71 requirement ideas, and for the other, the EPMcreate group generated 98 requirement ideas. To get a very crude estimate of what a group would do for FiemmeSkiJazz, multiply the average for an individual by 4 to get a number requirement ideas generated by a virtual group of 4. Four times the analysts' average of 24.86 is 99.44, and four times the musicians' average of 27.50 is 110, larger than either of the true group numbers, but in the same order of magnitude.

This calculation ignores overlapping ideas in a virtual group, self-management overhead in a real group, synergy within a real group, differences in the total real time available to both kinds of groups, and differences in the considered CBSs. Nevertheless, this simple calculation shows that the average number of total requirement ideas generated per individual in the current experiment was in the satisfactory range and allows saying that H1 is tentatively supported.

Hypothesis H2 is that the EPMcreate CET can be applied for requirements elicitation by a domain-expert end user. While support for H1 comes from the quantity of requirement ideas generated by the subjects, support for H2 must come from consideration of the quality of the requirement ideas. The Section 4.1 analysis of the ideas according to the 7 Loci Metamodel showed that the ideas were of an acceptable quality. This judgement was corroborated by an expert in the Web site's domain. The chosen domain expert was the manager of the local tourist office that was in charge also for marketing the festival. He had not participated in the experiment itself in order to avoid his possibly influencing the ideas generated. We gave the requirement ideas generated by the subjects to the chosen domain expert, and we received two types of feedback:

1. The first type of feedback was given verbally to one of the authors of this paper. This feedback gave an evaluation of "satisfactory" to all the generated requirement ideas. Apparently, the owner of the Web site found some of these ideas useful for solving the event's communication problems.

2. The second type of feedback was obtained implicitly by our and the Web master's determining how many of the generated requirement ideas had been implemented in the version of the Web site visible after the experiment was completed. All requirement ideas had been implemented in the new site, *except*
   - those that required changes in the organizational strategies;
   - some that required investments that were too high given (1) the limited budget of the event and (2) the limited return one could expect from their implementation;
   - some that were re-interpreted to address the trade-off between organizational effort and return.

That is, the subjects' ideas were considered good enough to be implemented.

5 Threats to Validity and Limitations

Besides the specific threats mentioned elsewhere in the paper, the experiment suffers from a number of threats including:

- its preliminary status,
- that we did not compare the individuals performing EPMcreate on the Web-site enhancement problem with some groups performing EPMcreate for the same amount of time on the same Web-site enhancement problem, as one might expect given the questions raised in the introduction.
- that only two kinds of stakeholders were used,
- that only two viewpoints were exercised in the sessions, and
- that one kind of CBS was used and that within the Web-site genre of CBSs, only one site was used.

As demonstrated in Section 3.14, the experiment actually achieves good construct validity with its quantitative and qualitative measures of the effectiveness of a CET simply because the good ideas according to these measures are the good ideas according to domain experts [8]. The shakiest measure the experiment uses is that for measuring the native creativity of the subjects, the Williams test [12]. The fundamental weaknesses of any psychometric test notwithstanding, the Williams test is accepted as testing native creativity, and in any case, has been used in several experiments with CETs precisely for this purpose [7, 11].

6 Conclusions

The preliminary experiment described in this paper was to begin answer two questions:

1. Can an individual use EPMcreate as well as a group does?
2. Can a domain-expert, end user use EPMcreate as well as a system analyst does?

The experiment involved 13 subjects divided into two kinds. The first kind of subject was an undergraduate Economics student studying a course about building quality Web sites, serving as an analyst. The second kind of subject was a musician, an expert in the domain of the FiemmeSkiJazz Web site. Each subject used EPMcreate to generate requirement ideas for the Web site. The results of the preliminary experiment allow giving a tentatively favorable answer to both questions. However, because of the threats to the validity of the experiment, similar experiments need to be carried on other applications, using only professional analysts, using other configurations of stakeholder viewpoints, etc. Finally, experiments need to be set up doing the full comparisons that the questions beg.

Acknowledgments

Daniel Berry's work was supported in parts by a Canadian NSERC grant NSERC-RGPIN227055-00 and by a Canadian NSERC–Scotia Bank Industrial Research Chair NSERC-IRCPJ365473-05. Luisa Mich's work was supported in part by a Canadian NSERC–Scotia Bank Industrial Research Chair NSERC-IRCPJ365473-05 and in part by the EU Papyrus project — ICT 215874.

References

1. Gause, D., Weinberg, G.: Exploring Requirements: Quality Before Design. Dorset House, New York, NY, USA (1989)
2. Maiden, N., Gizikis, A., Robertson, S.: Provoking creativity: Imagine what your requirements could be like. IEEE Software **21** (2004) 68–75
3. Nguyen, L., Shanks, G.: A framework for understanding creativity in requirements engineering. J. Information & Software Technology **51** (2009) 655–662
4. Osborn, A.: Applied Imagination. Charles Scribner's, New York, NY, USA (1953)
5. Aurum, A., Martin, E.: Requirements elicitation using solo brainstorming. In: Proc. 3rd Australian Conf. on Requirements Engineering, Deakin University, Australia (1998) 29–37
6. Jones, S., Lynch, P., Maiden, N., Lindstaedt, S.: Use and influence of creative ideas and requirements for a work-integrated learning system. In: Proc. 16th IEEE International Requirements Engineering Conference, RE'08, IEEE Computer Society (2008) 289–294
7. Mich, L., Anesi, C., Berry, D.M.: Applying a pragmatics-based creativity-fostering technique to requirements elicitation. Requirements Engineering J. **10** (2005) 262–274
8. Mich, L., Berry, D.M., Franch, M.: Classifying web-application requirement ideas generated using creativity fostering techniques according to a quality model for web applications. In: Proc. 12th Int. Workshop Requirements Engineering: Foundation for Software Quality, REFSQ'06. (2006)
9. MacCrimmon, K.R., Wagner, C.: Stimulating ideas through creativity software. Management Science **40** (1994) 1514–1532
10. Franch, M.: Marketing delle Destinazioni Turistiche. Metodi, Approcci e Strumenti. McGraw-Hill, Milano, Italy (2010) in Italian.
11. Sakhnini, V., Berry, D.M., Mich, L.: Validation of the effectiveness of an optimized epmcreate as an aid for creative requirements elicitation. In: Proc. 16th Int. Working Conf. Requirements Engineering: Foundation for Software Quality, REFSQ'10. (2010)
12. Williams, F., Taylor, C.W.: Instructional media and creativity. In: Proc. 6th Utah Creativity Research Conf., New York, NY, USA, Wiley (1966)

4 First International Workshop on Product Line Requirements Engineering and Quality (PLREQ)

Editors

Anil Kumar Thurimella
Harman International, AnilThurimella@gmail.com

Klaus Schmid
University of Hildesheim, schmid@sse.uni-hildesheim.de

Dirk Janzen
metadoc GmbH, di.janzen@web.de

Technical Programme

First International Workshop on Product Line Requirements Engineering and Quality (PLREQ'10)

Anil Kumar Thurimella[1], Klaus Schmid[2], Dirk Janzen[3]

[1] Harman International, Boecker-Goering-Str. 16, 76307 Karlsbad, Germany
AnilThurimella@gmail.com

[2] University of Hildesheim, Hildesheim, Germany
schmid@sse.uni-hildesheim.de

[3] metadoc GmbH, Birkenfeld, Germany
di.janzen@web.de

Abstract. The workshop focuses on quality aspects in requirements engineering for software product lines. Traditional approaches for quality requirements and variability management do not fully address the problems associated with quality requirements during variability modeling, product instantiation and product line evolution. In the workshop, novel approaches, emerging ideas and tools as well as industrial experiences to deal with qualities are discussed. Furthermore, the workshop will provide an interactive environment between researchers and practitioners.

1. Introduction

Quality in product line engineering is of major importance. In an organization with product lines, only good quality domain engineering artifacts are internally accepted by the engineers. Any quality flaws impact several products. However, this is also complicated by the fact that different products of a product line might be characterized by differences in quality requirements. Therefore, approaches should deal with variability in quality requirements as well. Recent advances have also shown that quality requirements effect the selection of artifacts during product instantiation.

The workshop provides a forum to discuss issues, novel approaches and tools within the area product line requirements engineering and quality engineering. Particular topics for the workshop include, but are not restricted to:

- Quality requirements and variability
- Infrastructures to improve quality in product line engineering
- Quality metrics for product lines
- Quality aspects of product line architecture

- Quality aspects in distributed product line engineering
- Empirical studies in the area of product line requirements engineering focusing on quality aspects
- Agile requirements engineering and quality
- Quality assurance in product line requirements engineering
- Application of data mining and machine learning techniques for software quality
- Impact of product line engineering on quality and vice versa
- Security requirements in product line engineering
- Project and process qualities

Program Committee

Ebrahim Bagheri, Athabasca University, Canada
Deepak Dhungana, LERO, Ireland
Jörg Dörr, Fraunhofer IESE, Germany
Paul Gruenbacher, University of Linz, Austria
Oystein Haugen, SINTEF, Norway
Mike Hinchey, LERO, Ireland
Robyn R. Lutz, Iowa State University, USA
T. Maruthi Padmaja, IDRBT, India
Juha Savolainen, Nokia, Finland
Christa Schwanninger, Siemens AG, Germany

2. Workshop papers

The workshop proceedings include four workshop papers as well as a workshop summary. The papers were presented in the workshop in the form of three sessions.

Session 1: Invited talks - Tools and Workflows

1. Kim Lauenroth, Invited Talk: Tool Support for Model-based Product Line Requirements Engineering - Challenges and Solution Ideas.

2. Arnaud Hubaux, Ebrahim Abbasi, Andreas Classen, Patrick Heymans, Workflow-driven Product Derivation.

Session 2: Decision-making

3. Jaap Kabbedijk, Krzysztof Wnuk, Bjorn Regnell and Sjaak Brinkkemper. What Decision Characteristics Influence Decision Making in Market-Driven Large-Scale Software Product Line Development?

Session 3: SPL Knowledge

4. Sebastian Adam, Joerg Doerr, Michael Ehresmann and Pascal Wenzel, Incorporating SPL Knowledge into a Requirements Process for Information Systems – An Architecturedriven Tailoring Approach

Invited Talk: Tool Support for Model-based Product Line Requirements Engineering - Challenges and Solution Ideas

Kim Lauenroth, André Heuer

Paluno – The Ruhr Institute for
Software Technology
University of Duisburg-Essen, 45127 Essen
{kim.lauenroth | andre.heuer}@paluno.uni-due.de

1 Introduction

The variability of a software product line is one of the main reasons for complexity in product line engineering [1]. To deal with this complexity, several researchers propose an orthogonal approach for variability modeling [2]. An orthogonal approach documents the variability of the product line in a separate model and the effects of the variability on the product line artifacts (e.g. requirements, design, or code) by means of relationships between the variability model and the product line artifacts.

2 Challenges

From a documentation-oriented point of view, the orthogonal variability modeling approach offers several benefits over other variability modeling approaches [3]. However, the visualization of the models and an effective visual modeling with the orthogonal approach constitute a significant challenge. An effective visualization and support for the creation of product line models (which we call visual modeling) is a crucial factor for the successful industrial application of such approaches, especially during requirements engineering.

In order to understand the meaning of effective visual modeling for requirements engineering, we present and discuss the following challenges for the visual modeling of requirements artifacts in product line engineering:

1. A support for the efficient creation and visualization of a variability model with 100+ variation points and variants is needed.
2. A support for the efficient definition of variability constraints in large and complex variability models is needed.
3. A support for a well-arranged visualization of constraining relations in large and complex variability models is needed.
4. A support for the visualization of the effects of constraining relations in large and complex variability models is needed.
5. The information whether product line requirements are common or variable are needed to be visualized.

3 Solution Ideas

In our presentation, we want to go a first step towards an effective visual modeling of product line requirements of complex software product lines based on an orthogonal variability modeling.

We present our prototypical tool implementation, named Remidemmi that addresses the abovementioned challenges. Remidemmi offers an editor for the orthogonal variability model and editors for the different types of requirements models. Remidemmi supports for example:

- Message sequence charts for the model based documentation of scenarios
- I/O-automata for the documentation of requirements related to the behavior of product line components.
- Temporal logic specification for the model based documentation of behavioral requirements.

Since the variability of the requirements models is documented in an orthogonal variability model, the variability model editor in our tool provides dedicated mechanisms for the creation and visualization of an orthogonal variability model. In this presentation, we will focus on two particular mechanisms.

The first mechanism is an abstraction layer mechanism that allows dividing the variability model into different abstraction layers. Each abstraction layer contains only a subset of the variability model. The second mechanism is a relationship exploring mechanism that allows exploring variability models that are too large to fit on a computer screen. Model elements that are too far away from a currently visualized and selected model element are shown on the border of the screen.

4 References

[1] Sinnema, M. et al.: Managing Variability in Software Product Families. In Proceedings of the 2nd Workshop on Software Variability Management, 2004.

[2] Bachmann, F. et al.: Managing Variability in Product Family Development. In Proceedings of 5th Intl. Workshop on Product Family Engineering (PFE-5), 2003.

[3] Pohl, K. et al.: Software Product Line Engineering – Foundations, Principles, and Techniques. Springer, Heidelberg, 2005.

Workflow-driven Product Derivation

Arnaud Hubaux, Ebrahim Abbasi, Andreas Classen, Patrick Heymans

PReCISE Research Centre, Faculty of Computer Science, University of Namur
Namur, Belgium
{ahu, eab, acs, phe}@info.fundp.ac.be

Variability models, feature diagrams (FDs) ahead, are commonly used to document product line (PL) requirements. They are also often used during product derivation where they are fed to configuration tools which support semi-automated feature selection. Configuration tools facilitate this task by automatically propagating the decisions made and by ensuring their overall consistency. However, most feature-based configuration tools assume that there exists a single monolithic FD and do not account for configuration processes that are distributed among various stakeholders who have specific concerns and who intervene at different moments. Our collaborations with industry have confirmed the need for techniques and tools that support such complex configuration processes.

In order to provide a modelling and reasoning framework for this process, we built upon our earlier work on formal semantics for FDs [1] and proposed *feature configuration workflows* (FCWs) [2], a formalism that combines the workflow language YAWL [3] with FDs.

An FCW, such as the one shown in Figure 1, is a workflow where tasks (such as Web Administrator or PloneMeeting Manager) are associated with FDs. In our work, we used YAWL as workflow modelling language. The configuration process follows the workflow. A FD is only configured when the task to which it is linked (through a start link) is executed. Generally, a task is assigned to a stakeholder, and the FD that is configured during the task captures her concerns and responsibilities. The second kind of node in a workflow is the condition (e.g. ▶, pm or ■) which designates a point in time. Linking a FD to a condition (through a stop link) means that the FD has to be fully configured when the condition is reached. In the example, it means that once pm is reached, there cannot be any decision left open in the FD of the web administrator and PloneMeeting manager.

The workflow is not the only link between FDs. Constraints can also be added across FDs, such as the $\ll excludes \gg$ link between features *Standard workflow* and *Archived* in Figure 1. Since the task and the stop of a FD are not necessarily directly linked to each other, the configuration decision can be postponed until the condition is reached. For instance, the stop of the Web Administrator, *viz.* pm, is placed only after PloneMeeting Manager. The decision of including or excluding the feature *Standard workflow*, for instance, does not have to be taken during task Web Administrator, because the person executing PloneMeeting Manager is also able to make this decision, the feature *Standard workflow* being the same.

The original implementation strategies for FCWs we studied are available in [2]. These strategies have been later completed to address FCW normalisation and decision

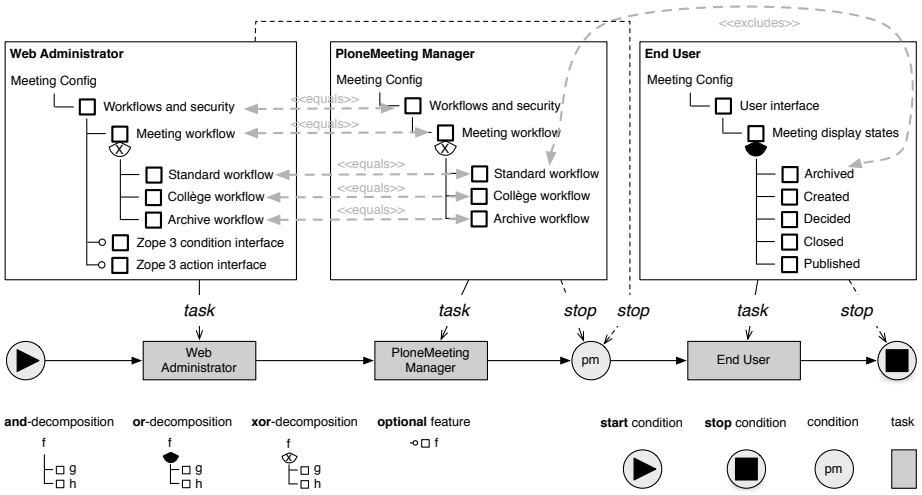


Fig. 1. Example FCW taken from the PloneMeeting configuration menu.

postponement problems [4]. Yet, an efficient implementation of these solutions is still to be provided.

In order to automate FCW editing and configuration, we are developing a tool based on YAWL and SPLOT [5]. YAWL provides support for workflow editing, task management and proposes a repertoire of advanced checks on workflows like soundness or weak soundness [6]. SPLOT provides support for FD configuration and constraint propagation [5]. Since YAWL and SPLOT are web-based applications, we are building a tool that relies on web-services to maintain the connection between the task management and FD configuration environments, and to enforce FCW semantics.

The next step on the agenda is to implement algorithms that evaluate whether an FCW is in *normal form* [4] and whether deadlocks can occur. In addition, the merging of inconsistent configurations obtained in concurrent environments (e.g. distributed offline configuration) will have to be dealt with.

Acknowledgements

This work is sponsored by the Interuniversity Attraction Poles Programme of the Belgian State, Belgian Science Policy, under the MoVES project and the FNRS.

References

1. Schobbens, P.Y., Heymans, P., Trigaux, J.C., Bontemps, Y.: Feature Diagrams: A Survey and A Formal Semantics. In: RE'06. (September 2006) 139–148
2. Hubaux, A., Classen, A., Heymans, P.: Formal modelling of feature configuration workflow. In: SPLC'09, San Francisco, CA, USA (2009)
3. van der Aalst, W., ter Hofstede, A.: Yawl: yet another workflow language. Information Systems **30**(4) (2005) 245–275

4. Classen, A., Hubaux, A., Heymans, P.: Analysis of feature configuration workflows (poster). In: Proceedings of the 17th IEEE International Requirements Engineering Conference (RE'09), Atlanta, Georgia, USA (2009)
5. Mendonça, M.: Splot. http://www.splot-research.org/ (May 2010)
6. van der Aalst, W.M.P., van Hee, K., ter Hofstede, A., Sidorova, N., Verbeek, H., Voorhoeve, M., Wynn, M.T.: Soundness of workflow nets:classification, decidability, and analysis. Technical report, Technische Universiteit Eindhoven (2008)

What Decision Characteristics Influence Decision Making in Market-Driven Large-Scale Software Product Line Development?

Jaap Kabbedijk[1], Krzysztof Wnuk[2], Bjorn Regnell[2], and Sjaak Brinkkemper[1]

[1] Department of Information and Computing Sciences
Utrecht University, Netherlands
{J.Kabbedijk, S.Brinkkemper}@cs.uu.nl
[2] Department of Computer Science
Lund University, Sweden
{Krzysztof.Wnuk, Bjorn.Regnell}@cs.lth.se

Abstract. Time efficiency is crucial for decision making in large scale market driven software product line development. In order to identify what factors influence the decision lead time and outcome, we conducted a retrospective case study at a large product software manufacturer and statistically analyzed seven possible relationships among decision characteristics. A large requirements engineering decision log was used to statistically test all hypotheses. The results show that the number of products affected by a decision has a positive relationship with the time needed to take a decision. Furthermore, more products imply a longer decision lead time. Results also show that when a change request originates from an important customer, the request is sooner accepted than changes requested internally. For efficient requirements management, our findings support that decision making activities can be carefully refined in large scale requirements engineering processes. Our findings, may be useful for Product Managers to understand the consequences of making certain types of decisions and planning actions in order to avoid their negative effects.

Keywords: Requirements Engineering; Decision Making; Market Driven Development; Software Product Lines;

1 Introduction

Requirements engineering is accepted as one of the most crucial stages in software design and development as it addresses the critical problem of designing the right software for the customer [3]. This critical problem gains even more importance in the Market-Driven Requirements Engineering mode, where the product content has to be aligned with the need of the targeted market segments, often estimated to thousands or millions of potential users, in order to create a profitable software product [24]. In order to decrease the cost and increase the ability to provide an individualized software product, the concept of

software platforms in combination with mass customization is often used [22]. This concept, called Software Product Lines [22] allows software development organizations to reuse a common base of the technology and, at the same time, to bring out products in close accordance with customers' wishes. The inevitable cost for a greater degree of reuse and increased productivity is an increased complexity of coexisting product variants and a more complex decision making process. In this complex environment, deciding which requirements to include into the scope of an upcoming project is not a trivial task. Moreover, the effects of certain scoping decisions may severely impede the quality of software products, for example when accepting significant changes to the product line late in the process as late changes usually require substantial effort to be held and have higher impact on the quality of products. On the other hand, time is a scarce resource in every business, so knowing exactly how much of this resource is needed for a certain decision or project is decisive [18]. As a result, the selection process may turn out to be a complex decision problem, where often sufficiently supportive techniques for assisting in this process based on cost-value approach, like for example prioritization [15], have to be extended to additional factors. What actually are these additional factors influencing both the time needed to make the decision (also called the decision lead time) as well as the outcome of the decision process? In this paper, we performed a retrospective analysis of the decision making process in a large-scale product line project with the aim of identifying which characteristics of changes may influence the decision lead time and the decision outcome. Our results can support software product managers in knowing which consequences are of certain characteristics of a decision. When a product manager is aware of certain consequences he can take adequate actions in order to avoid negative effects and by that contribute to the improvement of the requirements management process within the company.

Decision making is an important aspect of requirements engineering, which by some researchers is provocatively put in the center of the field [1, 2, 10]. A number of challenges in the requirements engineering decision-making (REDM) field has been defined [1, 16], stressing the need to understand which factors affect requirements engineering decision makers. Furthermore, the need for empirical studies of REDM has been stressed [2, 21, 1]. The distinction between diagnosis and look-ahead ways of supporting decision making is proposed by Pomerol [23], who focuses on supporting look-ahead decision making. Various techniques, even as advanced as the Constraint Satisfaction Problem Solution Techniques [9] have been proposed to automatically reduce the space of choices for ambiguities, for example in the software design decision process. The problem of selecting right requirements to the next project or product release has been described or addressed in a number of studies. Among them, Karlsson [15] promotes a cost-value approach to support this activity, later experimentally compared to other prioritization techniques [17]. Wohlin and Aurum [29] investigated the reasons for including features, while Wnuk et al. [28] investigated the reasons for excluding features from the scope of the project. The investigation of REDM in large scale bespoke development performed by Alenljung and Persson [1] confirms our

viewpoint of a large number of related aspects and dimensions of REDM that have to be considered in order to grasp its full complexity.

The paper is structured as follows. A description of the case company is given in Section 2. Our research design can be found in Section 3, together with the research questions in Section 3.2. After this, the statistical analysis of the decision logs will take place (Section 4), followed by the results (Section 5). The paper is ended by the conclusion and future research in Section 6.

2 Case Company Description

The results from the content analysis part of this paper are based on empirical data from an industrial project at a large company that is using a product line approach [22]. The company has more than 5000 employees and develops embedded systems for a global market. There are several consecutive releases of the platform, a common code base of the product line, where each of them is a basis for one or more products that reuse the platform's functionality and qualities. A major platform release has approximately a two year lead time from start to launch, and is focused on functionality growth and quality enhancements for a product portfolio. Minor platform releases are usually focused on the platform's adaptations to the different products that will be launched with different platform releases. The company uses a stage-gate model with several increments [5]. The scope of the release project is constantly changing during this process, from the initial roadmap extraction which is a basic for creating high level features to the final milestone of the requirements management process after which the development phase starts. In this case, the project management makes scoping decisions based on groups of requirements that constitute new functionality enhancements to the platform, called features. Change requests to these features are performed constantly by stakeholder from inside and outside the company. The scope of each project is maintained in a document called the Feature List, that is regularly updated each week after a meeting of the Change Control Board (CCB). The CCB exists of a permanent group of product and platform managers, complemented with other project stakeholders to a total of 20 members. The role of the CCB is to decide upon adding or removing features according to changes that happen.

Each change request to the scope of the project within the case company is registered in the CCB decision log. An example of an entry in the decision log is shown in Table 1. For reasons of confidentiality we used fictive data. This decision log comprises a number of attributes like: the change submitter and justification, the date that the request has been submitted, the decision date, the products impacted by a change, the release of the platform project impacted by a change, and the markets impacted by a change. For our research we were granted access to an extensive decision log with all data. The decision log of all products planned to be released in 2008 containing 1439 change requests was used as input for the content analysis presented in Section 4.

In the case company, a change request is filed after which, among others, ambiguity and completeness of the request are analyzed. This analysis is based on the Quality Gateway described by Natt och Dag et al.[19]. If the request is ambiguous or incomplete, it is sent back to the submitter to ask for a clarification, otherwise the request is put on the CCB agenda in order to perform the Impact Analysis (IA). An IA is performed by the appropriate Technical Groups (TGs) that elicit and specify high-level requirements for a special technical area, and Focus Groups (FGs) that design and develop previously defined functionality. After the IA the request is presented at a CCB meeting and the change request is decided upon. When an analysis performed by a certain group is not clear enough, extra information can be requested before the final decision is made. If the request is accepted, the change is implemented, else the submitter gets a rejection notification. For an overview of the process see Figure 1

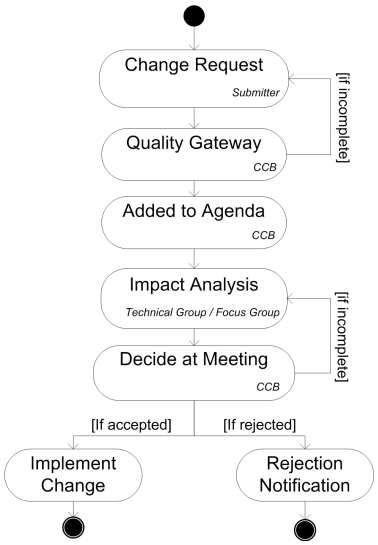


Fig. 1: CCB Decision Outline

ID	54
Change Request	HD resolution for video
Decision	Accepted
Comments	This will enlarge our market share in this sector
Release	Release 1.1
Description of change	Add HD resolution for recording
Justification	Requested by a large customer
Proposition Area	Video
Main affected TG	Video Group
Affected product	All products with camera
Affected key customer	Customer X
Affected FGs	HD Group
Submittal Date	09-02-09
RM tool ID	10F1
Decision Date	18-02-09

Table 1: Decision Log Entry Example

3 Research Design

In this section we will explain how we defined our variables and how the hypotheses and research questions are constructed.

3.1 Variables

The research we performed was mainly exploratory and done in order to first identify the main characteristics of decisions in REDM and second analyze the

relationships among these characteristics. After identifying the characteristics, we formulated research questions about relations within REDM and constructed our hypotheses based on these questions. We performed the appropriate statistical tests on the data from the decision log to either accept or reject all hypotheses and draw conclusions based on these test results. Based on the decision characteristics, five variables were created for each decision:

1. Lead Time: the duration between the moment a request was filed to the moment the decision was made by the CCB. The lead time is measured in week days and not working days, so there could by a small difference in days between two decisions who took the same number of working days to be taken, due to the weekend. As an example, about half of the decisions are made the same day they are requested (686 decisions, 48%), but the 753 requests that are left can take up to 143 days before a decision is made.

2. Number of Products Affected: a number between one and fourteen indicating the number of different products for which the requirements would change if the request was accepted.

3. Release Heartbeat: a variable strongly related to the release method used within the case company. As described in section 2, the product line platform of the case company is released in a heartbeat rhythm of one base release and four sequential releases. The release heartbeat variable indicates the specific number of the release affected by the change request. The higher the variable, the later the release is in the release heartbeat rhythm of the case company.

4. Customer: a nominal variable used to indicate whether a request is filed by an important external customer or is a request coming from inside the company. External customers in this case are very large partners of the case company who also help to bring the developed products to the market.

5. Decision Outcome: This variable indicates whether or not a change request is accepted by the CCB, it is also of nominal level of measurement.

3.2 Research Questions

Seven questions have been posed in this study in order to determine the relationship among different decision characteristics in requirements engineering decision making. According to Easterbrook et al. [8], in general all of these question are of the form "Are characteristic X and Y related?". All research questions are tested with a specific hypothesis that is, if possible, based on previous scientific work.

The first question (H^1) "Is the lead time related to the number of products affected" is based on Hogarth [13], who created a function on the relationship between the decision time and the task complexity. Hogarth states, based on mathematical models he created, that the amount of time needed to take a decision is an increasing function of the task complexity, till a certain point. After some point the costs of errors due to the task complexity becomes lower than the cost of time.

The question whether (H^2) the decision outcome is related to the number of products affected by a decision is also based on the work of Hogarth [13].

Since there is a tilting point in the relationship curve, there is a certain level of complexity, after which the decision maker decides the costs of errors due to a wrong decision are lower than the costs of spending any more time an making the decision.

Question three (H^3) concerning the relationship between the specific release heartbeat and the decision time and four (H^4) concerning the release and the decision outcome are partly based on the work of Saliu and Ruhe [25] and the work of Bagnall et al. [4], in which they suggest a relationship between decision outcomes and release planning. No explicit relationship between decision lead time or decision outcome and the release in a release cycle is claimed, but we expect to find such a relationship in our data.

Hallowell [12] empirically proved a relationship among customer satisfaction, loyalty and profitability. Because of this relationship, we believe it is reasonable to assume there is the possibility there is also a (H^6) relationship between the fact a request is filed by an important customer and the decision outcome. The case company benefits of keeping the customer satisfied and could because of this sooner accept requests of this customer than internal requests. The same reasoning goes for (H^5) the relationship between a request filed by an important customer and the decision lead time.

The last hypothesis formulated (H^7) is based on the work of Zur [30], in which he empirically proves a relationship between the time pressure people experience and the risks of their choice behavior. All these questions will be analyzed statistically in the next section.

4 Results

4.1 Test Selection

In order to perform parametric tests on the CCB decision log, all ratio level data should be distributed normally [11]. Since the variable "Lead Time" is the only variable of ratio level of measurement, we ensured this variable complied to the condition stated before. The variable "Lead Time" apparently described a log-normal distribution, so in order to be able to use this variable, the log_{10}-function of the variable was used for analysis. The D'Agostino-Pearson test [6] was used to see whether the log_{10}-function of the variable "Lead Time" described a Gaussian curve, or was distributed differently. With the D'Agostino-Pearson test we can test the following hypotheses (H^0):

$H^0_{0[1]}$:**The sample is [not] derived from a normally distributed population.**

When testing the kurtosis and skewness [7] of the distribution we found a result of $\chi^2(1, N = 753) = 35.3, p < .01$, which is below the critical value of 67.4 as can be found in the χ^2 distribution table. This means we can not reject H_0, so we can conclude that the log_{10}-function of the variable "Lead Time" is distributed normally and we can use parametric tests on this variable.

4.2 Effect of Number of Products Affected

We have two major hypotheses about the relationship of the number of products a decision affects. with other variables. The first hypothesis (H^1) on the relationship with the lead time of a decision can be described as:

$H^1_{0[1]}$: **The correlation between the number of products affected by a decision and the lead time needed to take the decision is [not] 0.**

Since we will analyze a possible relation between a variable of ration level of measurement and one of ordinal level of measurement, we used the non-parametric Spearman's Rank-Order Correlation Coefficient [27] to asses the correlation size. We found $\rho(752) = .222, p < .05$ after performing the test, which is higher than the listed critical value of .197 at a two-tailed level of significance of .05. This means we can reject H_0 and accept the hypothesis that the correlation between the number of affected products and the lead time is not 0.

The second hypothesis (H^2) we tested related to the number of products affected by a decision can be described as:

$H^2_{[0]1}$:**The number of products affected by a decision is [not] different for the different decision outcomes.**

Because the relationship between a variable of ordinal level and a variable of nominal level is tested, we use the Kolmogorov-Smirnov test for two independent samples [26]. We found a result of $Z = .545, p < 0.01$, which is higher than the reported critical value listed for Kolmogorov-Smirnov's Z at this level of significance. This means we can reject H_0 and accept our alternative hypothesis.

4.3 Effect of a certain Release

To test the effect of a certain release, we have stated two hypotheses. The first hypothesis about the effect of a certain release on the lead time of a decision (H^3) is as follows:

$H^3_{0[1]}$:**The correlation between the specific release heartbeat and the lead time needed to take the decision is [not] 0.**

To test this hypothesis about the correlation between a variable of ordinal level and a variable of ratio level, we used Spearman's Rank-Order Correlation Coefficient. The result of this test is $\rho(752) = .180, p < .05$, what is below the critical value of $\rho = .197$ for an $\alpha = .05$ two-tailed level of significance. This means we can not reject H_0 and we can not conclude there is any correlation between the release and the lead time needed to take a decision.

We also tested the relation between the release a decision affects and the decision outcome. We stated the following hypothesis (H^4):

$H^4_{[0]1}$: **The specific release heartbeat a decision affects is [not] different for the different decision outcomes.**

We used the Kolmogorov-Smirnov test for two independent samples for this analysis, which resulted in a score of $Z = 2.566, p < 0.01$. This result is well above the documented critical value of Kolmogorov-Smirnov's Z, what means we can reject H_0 and accept the alternative hypothesis H_1.

4.4 Effect of Large Customers

In this case, we first tested the difference of lead time needed to take decisions when large customers are involved in comparison with decision where they are not involved. In order to test this, we did a independent sample t-test on the following hypothesis (H^5):

$H^5_{[0]1}$: **The average lead time needed to take a decision is [not] different when a large customer is involved.**

The result of the t-test ($t(752) = .586, p = .558$) did not allow us to reject H_0, therefore we can state that based on our data there is no significant difference between the lead time needed to take decision when the decision is requested by a large customer, compared with a decision that is requested from within the company.

We also tested if there was any effect on the decision outcome caused by large customers. In order to test this we had to perform a χ^2 test for $r * c$ tables, because we tested the relation between two variables of nominal level of measurement. Our hypothesis (H^6) is:

$H^6_{[0]1}$: **The frequencies in the contingency table between the decision outcome and involvement of a large customer do [not] differ from the normal expected frequencies.**

The result of this test is with $\chi^2(1, N = 1439) = 7.032, p < .01$ above the listed critical value. This means we can reject H_0 and accept our alternative hypothesis. Since the value of χ^2 is rather low, we can state that the change of a positive decision outcome is with large likelihood a little higher when a decision is requested by a large customer.

4.5 Effect of Lead Time

In order to test whether the lead time influences the acceptance rate we stated the following hypothesis (H^7):

$H^7_{[0]1}$: **The average lead time needed to take a decision does [not] differ per decision outcome (i.e. accepted or rejected).**

The average lead time for rejected and accepted decisions is respectively $\mu = 1.12$ and $\mu = .98$. The result of the t-test ($t(752) = 3.940, p < 0.01$) indicated a significant differences between the average lead time for both decision outcomes. This means we can accept H_1 and reject the null-hypothesis.

5 Interpretation of Results

From the results presented in Section 4.2 we can see a significant relationship between the number of products affected by a decision and both the decision lead time (H^1) and outcome (H^2). Our results of testing on a large dataset can be interpreted as empirically confirming claims of Hogarth et al. [13], who state that the time needed to take a decision is highly dependent on the task complexity, in our case represented as number of products. To support out interpretation we have analyzed number of products involved in the decision and the decision lead time.

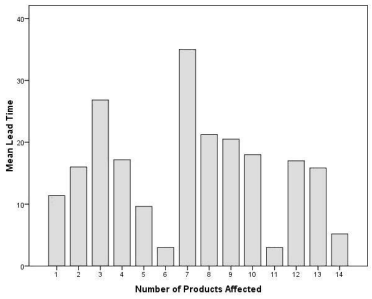


Fig. 2: Mean Lead Time per Number of Products Affected

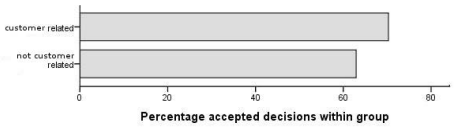


Fig. 3: Percentage of Accepted Decisions per Customer Type

Figure 2 shows an increase of the average lead time related to the number of products. If we compare the average lead time for 1 product with the highest lead time (for 7 products), the lead time becomes about five times longer. If we look at a more realistic comparison of lead time between the lead time for 1 product and 13 products we can still see an increase of 130% in average lead time. However, there appears to be no clear function to predict the amount of time needed to take a decision when the number of products is known, but there is a positive trend to be seen.

Other research [25, 4] also suggested a possible relationship between release planning and decision quality. This relationship could only partly be found in our case data, since there proved to be no significant relationship between the release a decision affects and the lead time (see Section 4.3). We did find a significant relationship between the release and the decision outcome (see Section 4.3), meaning decisions either get accepted or rejected more when the case company is later in their release cycle (H^4).

The fact that a request is filed by an important customers has no relationship with the decision lead time (see Section 4.4). It does however have a relationship with the decision outcome. Request filed by an important customer or more easily accepted than request coming from inside the company (H^6). We have further analyzed this result by analyzing the percentage of accepted decision

per customer type. The results are depicted in Figure 3, where we can see an 11% difference between the two groups. Expecting a higher acceptance rate on internal requests, because of an expected higher accuracy of internal requests, this relationship is remarkable.

The final result of our statistical analysis shows a significant relationship between the lead time and the decision outcome. This means that the decision outcome could be influenced by the time needed to take a decision. This implication could be of high relevance because it could mean that more wrong decisions are made in decision procedures that take a long time.

6 Conclusion and Future Work

Software Product Line scoping is a complex task, which includes analyzing many dependencies between customers and products derived from the product line in order to find an optimal set of features for a certain release. In Market-Driven SPL, the number of decision aspects that have to be taken under consideration and their dependencies grows significantly. In order to effectively improve RE decision-making we have to identify the key aspects of this process [1, 20]. Furthermore, the quality of decisions taken while deciding about the scope of the next release of software products directly influences the quality of the requirements for this release [1]. This in turn may improve the overall quality of software products.

In this paper, we performed a retrospective analysis of the decision making process in a large-scale product line project with the aim of identifying which characteristics of changes may influence the decision lead time and the decision outcome. Based on our case study statistical analysis, we can conclude that:

- There is a relationship between both the number of products affected in a decision and the time needed to take a decision and decision outcome. Our conclusion here is that decisions are sooner accepted when they have a large number of products they affect, than when they affect a lower number.
- Change requests done by an external customer are more likely to be accepted than internal request. Requests filed by an important customer have an 11% higher change to be accepted than other requests.

Our results provide valuable information for project manager that can be used to estimate the decision lead time for complex changes. In our case, the lead time turned out to be up to 400% longer if a decision affect multiple products. The fact that changes submitted by external customers are more likely to be accepted has two sides; on one hand it can be positive since Hallowell [12] and Kabbedijk et al. [14] both stated the importance of listening to customers in order to get their satisfaction and loyalty. On the other hand, this effect could also mean that change request filed by an important customer are only accepted to satisfy this customer and not because it is a useful change to the product. Product management processes can be adapted when being aware of the supported relationships.

Future research is needed to go more in depth on the possible relationships among REDM characteristics. Two relationships could be proven and quantified by us, based on the dataset, but the other five relationships need further research in order to further validate them. Within the two relationships proven by us, more research is needed as well. For instance, it would be helpful if a function could be formulated to estimate the lead time or the chance on a certain decision outcome. Finally, other decision characteristics, such as the number of stakeholders involved, could also be of relevance for the decision lead time or outcome.

Acknowledgments The authors would like to thank Thomas Olsson and Prof. Per Runeson for their suggestions in this research project.

References

1. Alenljung, B., Persson, A.: Portraying the practice of decision-making in requirements engineering: a case of large scale bespoke development. Requirements Engineering 13(4), 257–279 (2008)
2. Aurum, A., Wohlin, C.: The fundamental nature of requirements engineering activities as a decision-making process. Information and Software Technology 45(14), 945–954 (2003)
3. Aurum, A., Wohlin, C.: Engineering and managing software requirements. Springer Verlag (2005)
4. Bagnall, A.J., Rayward-Smith, V.J., Whittley, I.M.: The next release problem. Information and Software Technology 43(14), 883 – 890 (2001)
5. Cooper, R.: Stage-gate systems: a new tool for managing new products. Business Horizons 33(3), 44–54 (1990)
6. D'Agostino, R., Pearson, E.: Tests for departure from normality. empirical results for the distributions of $b^2 and \sqrt{b^1}$. Biometrika 60(3), 613 (1973)
7. DeCarlo, L.: On the meaning and use of kurtosis. Psychological Methods 2(3), 292–307 (1997)
8. Easterbrook, S., Singer, J., Storey, M., Damian, D.: Selecting empirical methods for software engineering research. Guide to Adv. Emp. Softw. Engineering (2007)
9. Egyed, A., Wile, D.S.: Support for Managing Design-Time Decisions. IEEE Transactions on Software Engineering 32(5), 299–314 (2006)
10. Evans, R., Park, S., Alberts, H.: Decisions not requirements decision-centered engineering of computer-based systems. In: The 1997 IEEE Conference and Workshop on Engineering of Computer-Based Systems. pp. 435–442 (1997)
11. Field, A.: Discovering statistics using SPSS. Sage Publications Ltd (2009)
12. Hallowell, R.: The relationships of customer satisfaction, customer loyalty, and profitability: an empirical study. International Journal of Service Industry Management 7(4), 27–42 (1996)
13. Hogarth, R.: Decision time as a function of task complexity. Utility, probability, and human decision making: selected proceedings of an interdisciplinary research conference, Rome, 3-6 September, 1973 pp. 321–338 (1975)
14. Kabbedijk, J., Brinkkemper, S., Jansen, S., van der Veldt, B.: Customer involvement in requirements management: Lessons from mass market software development. Proceeding of the 17th IEEE International Conference on Requirements Engineering pp. 281–286 (2009)

15. Karlsson, J., Ryan, K.: A cost-value approach for prioritizing requirements. IEEE software 14(5), 67–74 (1997)
16. Karlsson, L., Dahlstedt, A., Regnell, B., Natt och Dag, J., Persson, A.: Requirements engineering challenges in market-driven software development: An interview study with practitioners. Inf. and Softw. technology 49(6), 588–604 (2007)
17. Karlsson, L., T., T., Regnell, B., Berander, P., Wohlin, C.: Pair-wise comparisons versus planning game partitioning-experiments on requirements prioritisation techniques. Empirical Software Engineering 13(3), 3–33 (2007)
18. Leclerc, F., Schmitt, B., Dube, L.: Waiting time and decision making: is time like money? Journal of Consumer Research 22(1), 110 (1995)
19. Natt och Dag, J., Regnell, B., Carlshamre, P., Andersson, M., Karlsson, J.: Evaluating automated support for requirements similarity analysis in market-driven development. Proceeding of the 7th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'01) (2001)
20. Natt och Dag, J., Regnell, B., Gervasi, V., Brinkkemper, S.: A Linguistic-Engineering Approach to Large-Scale Requirements Management. IEEE Software 3, 32–39 (2005)
21. Ngo-The, A., Ruhe, G.: Engineering and Managing Software Requirements, chap. Decision Support in Requirements Engineering, pp. 267–286. Springer (2005)
22. Pohl, K., Böckle, G., Van Der Linden, F.: Software Product Line Engineering: Foundations, Principles, and Techniques. Springer-Verlag New York Inc (2005)
23. Pomerol, J.C.: Scenario Development and Practical Decision Making under Uncertainty: Application to Requirements Engineering. Requirements Engineering 3, 3–4 (1998)
24. Regnell, B., Brinkkemper, S.: Market-driven requirements engineering for software products. Engineering and managing software requirements pp. 287–308 (2005)
25. Saliu, O., Ruhe, G.: Supporting software release planning decisions for evolving systems. Proceedings of the 29th Annual IEEE/NASA Software Engineering Workshop pp. 14–26 (2005)
26. Smirnov, N.: On the estimation of the discrepancy between empirical curves of distribution for two independent samples. Bulletin Mathematics Univiversity Moscow 2, 3–14 (1939)
27. Spearman, C.: General intelligence: Objectively determined and measured. The American Journal of Psychology 15(2), 201–292 (1904)
28. Wnuk, K., Regnell, B., Karlsson, L.: What happened to our features? visualization and understanding of scope change dynamics in a large-scale industrial setting. In: Proceedings of the 17th IEEE International Requirements Engineering Conference (RE 2009). pp. 89–98 (2009)
29. Wohlin, C., Aurum, A.: What is important when deciding to include a software requirements in a project or release? In: Proceedings of the International Symposium on Empirical Software Engineering (ISESE 2005). pp. 246–255 (2005)
30. Zur, H., Breznitz, S.: The effect of time pressure on risky choice behavior. Acta Psychologica 47(2), 89–104 (1981)

Incorporating SPL Knowledge into a Requirements Process for Information Systems – An Architecture-driven Tailoring Approach

Sebastian Adam[1], Joerg Doerr[1], Michael Ehresmann[2], Pascal Wenzel[2]

[1] Fraunhofer IESE, Fraunhofer Platz 1, 67663 Kaiserslautern
{sebastian.adam, joerg.doerr}@iese.fraunhofer.de
[2] insiders technologies GmbH, Brüsseler Str. 1, 67657 Kaiserslautern
{m.ehresmann; p.wenzel}@insiders-technologies.de

Abstract. Software product lines (SPL) are a key concept for streamlining software development. However, building new information systems based on a SPL is often less efficient than expected because customer requirements often do not fit the capabilities or constraints of a reuse asset base. Especially in cases where SPL requirements are documented insufficiently, this problem is almost unsolvable. To cope with this challenge, a novel approach for flexibly aligning customer requirements with the capabilities of a SPL is therefore needed. Our idea is to tailor requirements processes based on the characteristics of a given reuse asset base without the need to reengineer all SPL requirements explicitly. To make this happen, an SPL-driven tailoring method that incorporates knowledge about a reuse asset base into a state-of-the-art requirements process is proposed in this paper. By applying this method in a case study, we have shown its feasibility and gathered initial experience regarding the suitability of the resulting requirements process.

1 Motivation

In order to faster develop customized information systems (IS), on which we focus in this paper,, many software enterprises have organized their development towards software product lines (SPL) in which the reusability of existing artifacts is potentially high [1]. Therefore, instead of developing each system from scratch, projects are often arranged around a reuse asset base that is customized (adapted, extended, or integrated) according to the specific needs of each individual customer. Besides assets from the company's own repository, components or services provided by third parties, e.g., open source modules, are also increasingly reused in this context.

However, even if such a development (often denoted as application engineering or product derivation) has been basically recognized as the key concept for gaining a competitive advantage, building new systems in this way is still a time-consuming and expensive task in practice [2]. Among others, one important reason for this low efficiency is caused by the ineffective and inefficient handling of customer requirements, which often results in mapping problems between customer needs and

SPL capabilities [3] [4], being especially a problem when SPL capabilities have not been documented sufficiently.

However, SPL requirements engineering [6], which aims at explicitly aligning usage requirements with reusable components, is not widely applicable for modern information systems, because an underlying assumption is often not fulfilled: In many cases, it is not economical to explicitly anticipate a satisfactory number of usage requirements, such as business process variants, or legacy interfaces during domain analysis, respectively scoping [14], in advance. While the variation points may be defined indeed, the concrete variants which are realizable based on the SPL can often therefore not be determined in advance. Furthermore, a simple resolution of a variability model alone is often not sufficient, because information systems have to satisfy very specific needs that must be systematically elicited first, Hence, reconciling customer requirements with the capabilities and constraints posed by a SPL's reuse asset base remains a huge challenge and unsystematic task today, as no integrated handling of the requirements that are expressible with predefined variability models (e.g., use case variants) and those that are customer-specific exists yet. As a consequence, many rework or tuning iterations are typically needed (e.g., for renegotiations, completions, and changes), which makes application engineering still less efficient than expected [8].

Aligning individual customer requirements with the capabilities of a certain SPL much earlier, but without the need to explicitly prescribe all feasible requirements (mainly usage requirements such as use cases or workflows) in advance, is therefore expected to be a fruitful means for increasing efficiency in SPL-based information system development. Hence, a novel mechanism in which the actual needs of customers are used as a starting point for requirements elicitation, but in which SPL characteristics are also taken into consideration when refining these requirements, must be found. Only when SPL characteristics are sufficiently known during the early requirements phase already, the chance that requirements fit the reuse asset base better and do not need to be renegotiated or completed in costly rework cycles increase [17].

So far, however, the impact of a reuse asset base and especially the SPL architecture on unforeseen requirements or vice versa can only be assessed by experienced people during elicitation. A systematic check of the fit in an analytical manner and improving the requirements afterwards is still the state-of-the-art.

This paper therefore proposes to tailor the application engineering requirements processes according to the characteristics of a reuse asset base. In contrast to existing SPL approaches, the actual capabilities of a SPL are therefore determined based on the detailed SPL design and not only based on early decisions of domain requirements engineering, It is therefore expected that a higher fit can be exploited when proceeding this way, and that, in particular, customer actual needs can be met better. The remainder of this paper is structured as follows: In section 2, related work is presented, while section 3 introduces our tailoring method for the incorporation of reuse knowledge into a requirements process. Section 4 describes our feasibility study while section 5 summarizes the paper.

2 Related Work

While much effort has been spent on how to build up reuse asset bases, the actual reuse during application engineering has not received sufficient attention yet [8] [9] [10]. Nevertheless, the impact of reuse on requirements engineering has been studied for many years.

Requirements engineering for COTS-based projects, for instance, deals with COTS selection and COTS adaptation [18] [28]. While the purpose of COTS selection is to find an existing (market) product that is closest to the customer's requirements, COTS adaptation [27] deals with the resolution of conflicts that might still exist between the actual customer needs and the features provided by the chosen product. However, with regard to the underlying problem of this paper, COTS-based requirements engineering approaches are not sufficient because they do not give any guidance as to which requirements are actually to be elicited in order to select and adapt reusable components efficiently. Furthermore, COTS-based RE rather deals with the reuse of one "best-fitting" market product than with the elicitation of requirements on a system to be built in a reuse-based way.

Requirements engineering for software product lines can be rather suitable for this purpose. For our purpose, however, only the requirements engineering in application engineering (AERE) is of interest. Within AERE, one can distinguish activities to instantiate variable requirements that were created during domain engineering already, and activities to elicit new requirements that have not been covered in domain engineering yet. However, even if the usage of both activities is important, recent approaches such as [23] or [26] only focus on the instantiation of variability, e.g., by picking features from a catalog. Even if this is a very efficient approach, it is only applicable in well-predictable domains because it relies on the restrictive assumption that all requirements and their dependencies can be explicitly prescribed in the domain engineering phase (which is otherwise neither economical nor feasible [2]). Thus, most of the highly automated product line approaches are very inflexible when to be used for customer-specific requirements [9]. This is especially a problem in modern information systems, which have to reflect individual business processes and which must often be integrated with proprietary and third party applications on the customer side. Hence, even for requirements that differ only slightly from the foreseen variants, fully automated product derivation approaches are not applicable any more [20], which results in manual, typically not guided, and thus costly extensions. [9] and [20] therefore propose an approach that systematically allows deriving products that are not explicitly foreseen but close enough to the product line core assets. Their idea is to replace explicit decision models by restricted transformation rules. However, their approach only addresses the actual product instantiation and gives no answer on how to elicit and negotiate the corresponding customer requirements in a systematic manner.

For eliciting requirements that are not covered by a product line, only some initial work exists [10]. So far, mostly the tasks of communicating the product line variability [17], selecting variants, specifying system requirements, and supporting trade-off decisions have been proposed as being important in this context [22]. [7] describe a detailed scenario-based approach and also [10] introduce an approach for more systematic AERE. However, one remaining problem is the fact that

requirements are always identified in a rather solution-driven than problem-driven way [21], as the aim is rather on a large degree of direct requirements reuse than on the satisfaction of unforeseen requirements with available solution components. This might be a risk, because customers typically know what they want, but not what they really need [13]. Hence, when high individuality is required in order to satisfy a customer, much stronger orientation towards real requirements becomes necessary. However, systematic guidance on how to elicit real needs and reconcile them with SPL capabilities is not supported systematically yet, and unaligned approaches such as proposed in [21] make a sufficient fit without costly rework almost impossible. So far, the problems mentioned in the motivation cannot be solved satisfactorily yet with current product line approaches.

Tailoring requirements processes towards actual reuse capabilities and constraints without the need to explicitly reengineer requirements is probably the only means currently able to solve the problem of having a gap between unforeseen customer requirements and the capabilities of a given reuse asset base. Indeed, the work of [19], aimed at improving requirements engineering processes, is maybe one of a few existing approaches that explicitly captures the specific needs and constraints of different project stakeholders. However, this approach lacks both systematic (non-brainstorming-driven) identification of the relevant requirements classes and an algorithmic reflection of these information needs in a requirements process. Furthermore, the context of reuse orientation is not considered. Unfortunately, also more recent work dealing with information needs such as [29], does not offer systematic guidance on how to derive these needs.

3 Incorporating Reuse Knowledge into a Requirements Process

In order to proactively guide the requirements process towards fitting customer requirements more efficiently, explicit knowledge about the capabilities, constraints, and needs posed by a SPL must be made available in the requirements process. A first step towards this goal is the early consideration of existing solution assets (not any requirements!) when defining requirements with customers [11]. However, as this notion in isolation does not give any guidance on how to perform the elicitation in an effective and efficient way, it is necessary to give requirements engineers very precise guidance in terms of an elicitation guideline that clarifies which requirements classes are to be discussed in which sequence and which constraints have to be considered. The main benefit of such an explicit guideline is that requirements engineers “know” which information they have to elicit (and which not) and which valid range of values can be provided without contravening the capabilities of the reuse asset base. Hence, omissions as well as superfluous elicitations can be avoided proactively, and negotiations can be initiated and finished successfully much earlier when customer requirements contradict the capabilities or constraints. Nevertheless, it has to be noted that the elicitation guidelines are not intended to restrict the creativity of customers but just answering prepared questions. Rather, the guideline should help the requirements engineer in performing a meaningful and especially complete elicitation.

In this context, the main difference of our proposal to other SPL requirements approaches is that already existing SPL characteristics are just used in order to give customers the possibility of aligning their real needs with innovative features on their own. Thus, real customer needs and expectations are, as in traditional RE, taken as a driver for the elicitation process, but knowledge about the SPL is used as a guide rail to prevent requirements engineers from going astray. However, this notion can only be successful if both the elicitation guidelines and the SPL capability representations are derived systematically based on the given reuse asset base and the corresponding developers' information needs. This section therefore describes our SPL-driven tailoring method for the systematic adaptation of requirements processes.

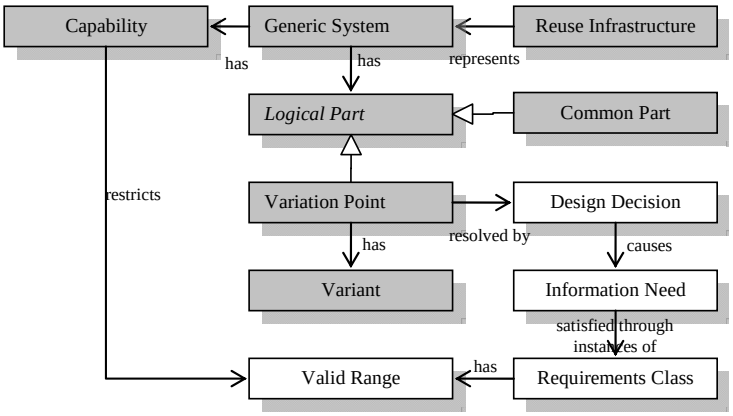


Figure 1. Decisions to be made during tailoring

3.1. Foundation for SPL-driven RE Tailoring

Before introducing the activities to be carried out during tailoring, the conceptual elements to be handled must be clarified first (see Figure 1). According to this model, which partially includes concepts of [16], the generic systems represented by a given reuse asset base are in the center of attention. A generic system provides certain capabilities depending on the logical parts it contains. In this context, logical parts are all runtime-relevant artifacts (independent of their granularity) that implement system capabilities, such as components, classes, sub-systems, configuration files, functions, interfaces, and the like. A logical part can either be common, optional, or variable within concrete instantiations of the generic system.

Logical parts that are variable or optional are called variation points and may have different variants that implement this variation point in alternative ways. Which variant is chosen in a concrete system depends on an (either explicit or implicit) design decision. In order to make this decision in a well-informed manner, specific information must be known. Hence, each decision point causes information needs (i.e., something an application engineer must know) that must be satisfied by a set of requirements. The simplest information need (which is often the one relevant in

traditional product line engineering) is the need to know which variant a customer wants. But as customers typically know what they want, but not what they really need [13], decision-making is usually a more complex process requiring additional information about the system context and the intended use. Therefore, appropriate requirements classes (i.e., types of requirements that are concerned with a specific issue, for instance, use cases) must be found whose instances are able to satisfy the given information needs. For example, deciding which database system should be used depends on information about available budget, required power of queries, or the amount of data to be stored,, which can be delivered by requirements classes such as "Project constraints", "Desired queries" or "Quantified data models". Hence, requirements delivering this information must be elicited explicitly during the requirements process. However, as the supported capabilities, especially those provided by different variants, may also pose constraints for the definition of realizable requirements, the valid ranges of the corresponding requirements classes have to be restricted accordingly.

3.2. Tailoring Phases

Based on the above-mentioned foundation, we developed a precise three-phase tailoring method that systematically maps the properties of a given reuse asset base to a requirements process by means of a systematic, repeatable, and efficient elicitation guideline. While Figure 2 shows the essential steps and artifacts of tailoring, the details of each phase are described below.

Infrastructure Analysis & Rescoping. The purpose of the infrastructure analysis & rescoping phase is the identification of existing reuse assets and their capabilities (as shown in the form of the gray boxes in Figure 1). Below, the activities of this phase are described.

1) Identify and classify logical parts. In the first step, the tailoring expert and the SPL architects analyze the SPL's reuse asset base (see (1) in Figure 2). The purpose of this step is to decompose the generic systems addressed by the SPL into their logical parts, and to decide whether these parts are mandatory or optional, respectively fixed or variable within these generic systems. A helpful starting point for this decomposition are the main layers of modern information systems. As a result of the recursive decomposition, a classified list of logical system parts is then obtained.

2) Determine valid variants and capabilities. Based on the list of logical system parts, the tailoring expert and the architects check in which ways the variable system parts (i.e., the variation points) can be currently implemented. Hence, for each variation point, different variants that are valid (i.e., already in scope) are explicitly listed or implicitly described in a list of valid variants, respectively a list of valid ranges of values. Then, these variants as well as the fixed parts are taken to extract all functional and non-functional capabilities of the entire generic system into a structured list of capabilities.

3) Determine adherence to capabilities. The tailoring expert, the product manager, and the architects then cooperatively decide whether they would like to insist on the available capabilities only during applications or whether they would like to allow modifications and extensions. If deviations are possible, boundaries of these deviations as well as criticality and operational reactions in case of deviation are defined in terms of "allowed deviations". The difference between allowed deviations and variants is that allowed deviations are possible but have not been explicitly anticipated, while variants have been.

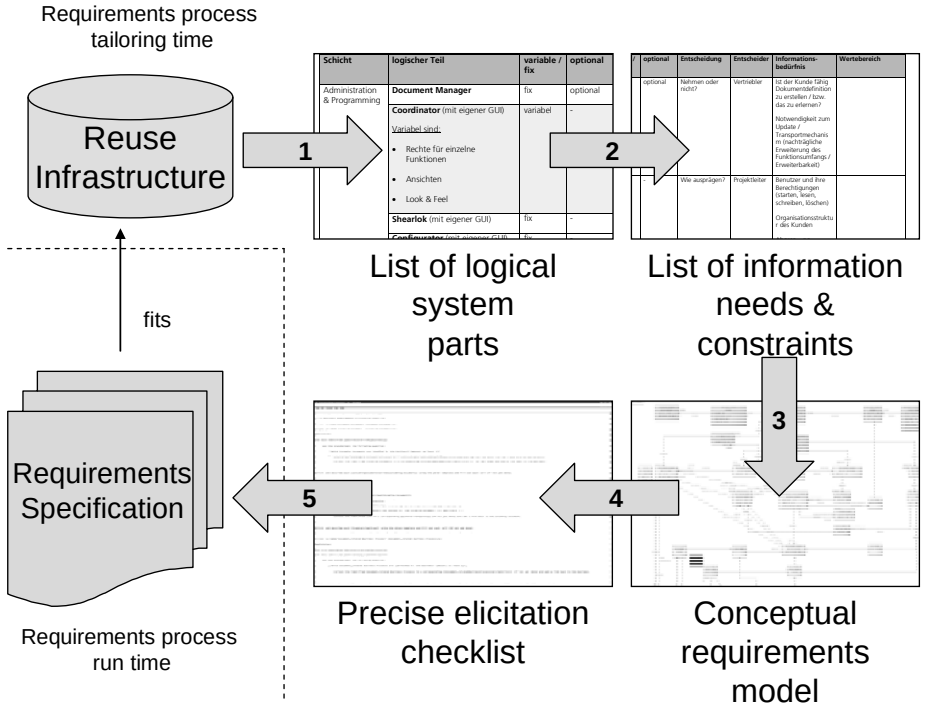


Figure 2. Essential tailoring steps and artifacts

<u>Information Need Analysis & Requirements Tailoring.</u> In the information need analysis & requirements tailoring phase, which is the core of our SPL-driven tailoring method, the analyzed knowledge about the reuse asset base and the adherence to its capabilities is taken to identify the requirements classes that are necessary for making relevant development decisions. The purpose of this phase is the systematic identification of information needs and SPL constraints that must be satisfied by, respectively considered during, a requirements process (see white boxes in Figure 1). Below, the activities of this phase are described.

1) Identify information needs. In this step (see (2) in Figure 2), the tailoring expert first identifies all people who are responsible for resolving variation points during application engineering. The tailoring expert then discusses with these people

each variation point they are responsible for in order to find out which information is needed for deciding how the variation points should be resolved. Furthermore, for each information need, valid ranges of values are defined based on the currently supported SPL capabilities and the allowed deviations. As a result of this second activity, a consolidated list of information needs including their valid ranges of values is obtained.

2) Incorporate information needs. During this step (see (3) in Figure 2), the requirements classes whose instances are able to satisfy the identified information needs are determined. For this purpose, a conceptual requirements model is created by the tailoring expert. This model describes all important requirements classes that must be discussed during the requirements process. Hence, as a result of this step, a model that formally expresses the actual information needs and their dependencies is obtained (see Figure 3 for an exemplary extract).

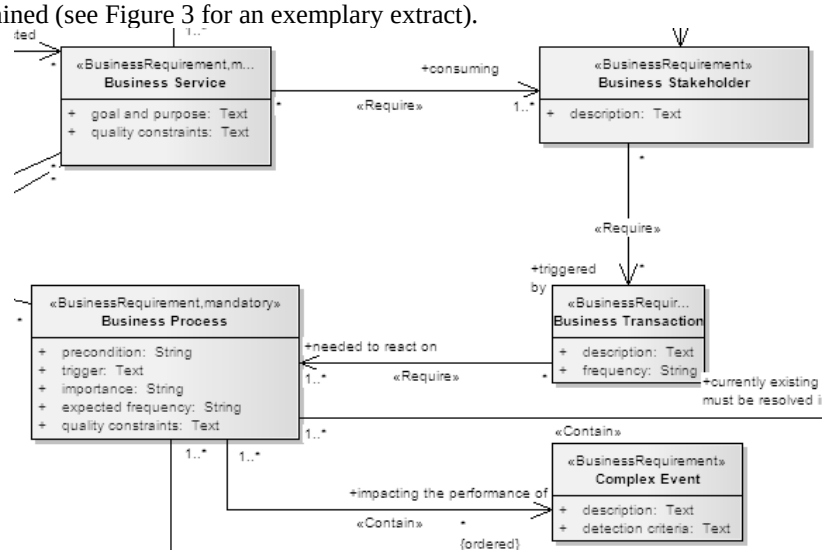


Figure 3. Extract of a conceptual requirements model

3) Incorporate reuse strategy. In order to reflect not only the information needs but also the reuse constraints and capabilities, the tailoring expert and the product manager decide which requirements classes should be restricted, either through attribute constraints or through explicit, predefined instances (e.g., variants). To make this decision, also real-world possibilities such as "Is it realistic to predefine all possible instances in advance?" have to be considered. As a result, the degree of freedom for each requirements class is reflected in the conceptual requirements model using tagged values. Then, the tailoring expert determines the concrete valid ranges of values for each requirements class, respectively their attributes, based on the previous decision. For each requirements class whose instances should not be predefined, the constraints described in the list of information needs are transferred to the conceptual requirements model. Then, for each requirements class whose instances are to be predefined, concrete instances are developed together with the product manager according to the structure and relationships of the requirements class. For this purpose, the already existing reusable requirements are taken into consideration.

Hence, as a result, an updated version of the conceptual requirements model and, if applicable, a set of predefined requirements are obtained.

Process Operationalization. The final process operationalization phase deals with rather operative decisions. The purpose of this phase is the implementation of the tailored requirements process in an organization, including the decision about which notations are to be used or in which form the reuse capabilities should be represented to the customers. Important sub-steps are the setup of a reuse repository and the generation of elicitation guidelines based on the conceptual requirements model developed before. Especially the latter model transformation (for technical details, see [24]) allows achieving high preciseness and individuality of the requirements process without any costly "handbook creation". Thus, organizations applying our tailoring method can benefit immediately from a concrete elicitation guideline.

1) Determine capability representations. As sales knowledge is a valuable input for product definition (or even derivation) [8], this activity deals with the decision on how to "sell" the reuse capabilities to a customer during the requirements elicitation. As the prediction of concrete requirements is limited in the information system domain, the tailoring expert and the product manager therefore determine how to represent existing solution components. This representation should, in particular, allow requirements engineers to guide the elicitation process towards realizable requirements when concrete requirements instances cannot be predicted in advance. Hence, the determination and creation of conceptual service descriptions [11] adequately presenting the reuse capabilities is basically the purpose of this step.

2) Generate guidelines and templates. In order to actually reflect all tailoring decisions in a requirements process, prescriptive and precise elicitation guidelines and specification templates should be used. Therefore, elicitation guidelines and specification templates are automatically generated based on the information needs and constraints expressed in the conceptual requirements model (see (4) in Figure 2). An exemplary extract of an elicitation guideline that describes instructions for the elicitation of a requirements class "Incoming Document" is shown in Figure 4.

```
For each identified [[BusinessOverview][Business]]:

    Ask the stakeholders the following question:

        _"Which Incoming Documents are  handled in  the Business? (Amount: at least 1)"_

            Collect the identified Incoming Documents in a corresponding [[IncomingDocumentOverview][list]] (if not yet


Elicit and describe each [[IncomingDocumentOverview][Incoming Document]] using the given template and fill out each cell
```

Figure 4. Extract from a precise elicitation guideline in TWiki syntax

3) Fill reuse repository. In this step, the tailoring expert imports and categorizes the predefined requirements developed in the previous phase into a repository using the generated templates. Furthermore, the conceptual services developed before are also stored in the repository, and additional tool, e.g., for specification is set up.

4 Feasibility of Tailoring

To investigate the feasibility of our SPL-driven tailoring method, we performed a case study in a medium-sized enterprise that develops content recognition software in the domain of invoice processing. Even if it is not possible to show all case study results in detail due to confidential reasons, the extracts shown in Figure 3 and Figure 4 are taken from real artifacts produced during the case study.

Background. Before the requirements process was tailored to its specific needs, the case study organization was not satisfied with its performance when developing customer-specific systems. The concrete problem this organization had was that the requirements specifications delivered by external sales partners were often not appropriate for application engineering, because application engineering-relevant information was missing, or requirements that could not be realized in time and within budget were promised to the customer. Hence, the project team was often faced with the need to renegotiate requirements or to implement features from scratch.

Objectives of Study. The goal of our case study was to analyze our SPL-driven tailoring method for the purpose of characterization with regard to feasibility and suitability from the viewpoint of the method engineers in the context of a SPL organization. The hypotheses we wanted to support were:

H1. Our tailoring method can be carried out as described above.

H2. Our tailoring method and its results are suitable for the context of IS developing organizations, i.e., a) the effort to perform the tailoring is justifiable, and b) the resulting elicitation guideline is applicable in real projects.

Data Collection and Analysis. To gather data with regard to the first hypothesis, we wrote down our experiences during tailoring. We checked whether our method steps could be carried out as planned or whether we had to change the way a step was performed in order to keep it applicable. To gather data with regard to the second hypothesis, we objectively measured the effort needed to perform each tailoring step. Furthermore, at the end of the case study, we let the persons who participated in the tailoring sessions fill out a questionnaire in order to get an impression of the practical applicability of the tailoring method and the resulting elicitation guideline.

Validity Threats. The case study has only shown basic possibilities of our tailoring method without generalizable results. A specific threat was the fact that we did not involve the real decision makers but only two surrogates from the case study organization. A further threat was that we carried out the tailoring by our own.

Case Study Performance. During the case study, we performed the SPL-driven tailoring as mentioned in section 3. The role of the tailoring expert was taken by the method developer, while two experts from the case study organization assumed all other roles. We organized the tailoring in three half-day workshops followed by subsequent "homework". The resulting documents were then discussed and reviewed by our contact persons in the case study organization.

Results. The case study confirmed the feasibility of our SPL-driven tailoring method. We could show that it is possible to define a tailored requirements process, respectively a precise elicitation guideline, based on a given reuse asset base when following our method. Therefore, our hypothesis *H1* (feasibility) has been strongly supported by the study.

With regard to *H2a* (suitability of method), we measured a total effort of 7.8 person-days (pd) with a ratio of 0.3 person-days per (high-level) system part. With 0.9 person-days, the "Incorporate information needs" was the most costly step.

In this regard, an interesting feedback was that the elicitation guideline could be developed "by only answering some questions". Hence, the people from our case study organization were impressed that they did not need to have deep requirements engineering experience when participating in the requirements process tailoring.

With regard to *H2b* (suitability of tailoring results), the achieved elicitation guideline was assessed as being very helpful for the case study organization. The participants were convinced that this guideline can support achieving a better fit between requirements and available components, leading to better quality of the requirements specifications and less effort in subsequent development phases. In particular, they mentioned that this guideline is a helpful tool for requirements engineers without making requirements elicitation a too rigid task. Furthermore, they emphasized that the side effects of our approach were extremely low in contrast to other reuse strategies: Apart from the guidelines used for the requirements process, nothing has changed in the organization, which makes this approach especially interesting for enterprises that have neither time nor money to invest in heavy-weight changes. Another side effect was the fact that inexperienced people can also be guided well in performing requirements elicitations by following the guideline. Hence, it is expected that elicitation can become less dependent on people doing it.

Limitations. Due to time restrictions, we have not applied the resulting elicitation guideline in real projects. So far, the results of our tailoring method have only been evaluated by subjective feedback but not through real use. Furthermore, we are aware that our approach is not the best choice when customer requirements can be predicted in advance to a large degree (in this case, a traditional product line should better be (re)engineered).

Open Issues. We have to check whether the tailoring results are reproducible. Furthermore, of course, we have to show that using the tailored guidelines actually results in a better fit between customer requirements and infrastructure capabilities. We are preparing a controlled experiment and a second case study in this regard.

5 Conclusion & Outlook

Building new information systems based on a SPL is often less efficient than expected because requirements often do not fit the characteristics of a given reuse asset base. Even if product line requirements engineering approaches explicitly aim at improving this fit, they often do not provide the required flexibility or tend to lose their systematic support when they are to be used for requirements that were not been foreseen during domain analysis already [21].

In this paper, an approach for reconciling customer requirements with the capabilities of a reuse asset base in more flexible way has therefore been motivated: Instead of confronting customers with predefined requirements, customer's real needs should be aligned with reuse capabilities during elicitation. To make this happen, the contribution of this paper is an SPL-driven tailoring method that incorporates

knowledge about a reuse asset base into a precise elicitation guideline without the need to explicitly reengineer requirements. The approach has been evaluated in an organization that provides highly individual document recognition systems based on a core product platform. The study has shown that our SPL-driven tailoring is feasible and that the resulting artifacts promise many benefits.

For the future, we plan to conduct additional validations and we are going to provide more guidance for the tailoring steps. Furthermore, we are going to improve the tool support and automatic generation of the process guidelines.

References

1. Lam, W., Jones, S., Britton, C.: Technology Transfer for Reuse: A Management Model and Process Improvement Framework. In: Requirements Engineering Conference. IEEE, 1998
2. Deelstra, S., Sinnema, M, Bosch, J.: Product derivation in software product families: a case study. In: The Journal of Systems and Software, vol. 74. Elsevier, 2005
3. O'Leary, P., Rabiser, R., Richardson, I., Thiel, S.: Important Issues and Key Activities in Product Derivation: Experiences from Two Independent Research Projects. In: Software Product Line Conference. SEI, 2009
4. Rabiser, R., Grünbacher, P. Dhungana, D.: Requirements for product derivation support: Resuts from a systematic literature review and an expert survey. In: Information and Software Technology. Elsevier, 2009
5. Baum, L., Becker, M., Geyer, L., Molter, G.: Mapping Requirements to Reusable Components using Design Spaces. In: Requirements Engineering Conference. IEEE, 2000
6. Clements, P., Northrop, L.: Software Product Lines. Addison Wesley, 2001
7. Bühne, S., Halmans, G., Lauenroth, K., Pohl, K.: Scenario-Based Application Requirements Engineering: In: Software Product Lines. Springer, 2006
8. Rabiser, R., Grünbacher, P., Dhungana, D.: Supporting Product Derivation by Adapting and Augmenting Variability Models. In: Software Product Line Conference. IEEE, 2007
9. Perrouin, G., Klein, J., Guelfi, N., Jezequel, J.: Reconciling Automation and Flexibility in Product Derivation. In: Software Product Line Conference. IEEE, 2008
10. Rabiser, R., Dhungana, D.: Integrated Support for Product Configuration and Requirements Engineering in Product Derivation. In: Conference on Software Engineering and Advanced Applications. IEEE, 2007
11. Adam, S., Uenalan, O., Riegel, N., Kerkow, D.: IT Capability-Based Business Process Design with Service-Oriented Requirements Engineering. In: LNBIP 29. Springer, 2009
12. Sommerville, I., Sawyer, P.: Requirements Engineering – A good practice guide. John Wiley, 1997
13. Davis, A.: Software Requirements – Objects, Functions & States. Prentice Hall PTR, 1993
14. Schmid, K.: Planning Software Reuse - A Disciplined Scoping Approach for Software Product Lines. PhD Theses in Experimental Software Engineering 12. Fraunhofer, 2003
15. Bayer, J., Flege, O., Knauber, P., Laqua, R., Muthig, D., Schmid, K., Widen, T.: PuLSE – Product Line Software Engineering. IESE-Report No. 020.99/E. Fraunhofer IESE; 1999
16. Muthig, D., Atkinson, C.: Model-Driven Product Line Architectures. In: Software Product Line Conference. Springer, 2002
17. Halmans, G., Pohl, K.: Communicating the variability of a software-product family to customers. In: Software and System Modeling 2003/2. Springer, 2003
18. Alves, C.: COTS-Based Requirements Engineering. In: Component-Based Software Quality, LNCS 2693. Springer, 2003

19. Doerr, J., Paech, B., Koehler, M.: Requirements Engineering Process Improvement Based on an Information Model. In: Requirements Engineering Conference. IEEE, 2004
20. Guelfi, N., Perrouin, G.: A Flexible Requirements Analysis Approach for Software Product Lines. In: Proceedings of RefSQ 2007, LNCS 4542. Springer, 2007
21. Djebbi, O., Salinesi, C.: RED-PL, a Method for Deriving Product Requirements from a Product Line Requirements Model. In: CAiSE 2007. Springer, 2007
22. Pohl, K.: Requirements Engineering – Grundlagen, Prinzipien, Techniken. dpunkt, 2007
23. Sinnema, M., Deelstra, S., Hoekstra, P.: The COVAMOF Derivation Process. In: Conference on Software Reuse. Springer, 2006
24. Yaco, M.: Transforming Information Needs into a Requirements Engineering Process, Master Thesis in Computer Science. TU Kaiserslautern, 2009
25. Lauesen, S.: COTS tender and integration requirements. In: Requirements Engineering, Vol. 11. Springer, 2006
26. Gomaa, H., Saleh, M.: Feature Driven Dynamic Customization of Software Product Lines. In: Conference on Software Reuse. Springer, 2006
27. Mohamed, A., Ruhe, G., Eberlein, A.: MiHOS: an approach to support handling the mismatches between system requirements and COTS products. In: Requirements Engineering, Vol. 12. Springer, 2007
28. Alves, C., Franch, X., Carvallo, J., Finkelstein, A.: Using Goals and Quality Models to Support the Matching Analysis During COTS Selection. In: LNCS 3415. Springer, 2005
29. Sommerville, I., Lock, R., Storer, T., Dobson, J.: Deriving Information Requirements from Responsibility Models. In: CAiSE 2009, LNCS 5565. Springer, 2009

Workshop summary: Product Line Requirements Engineering and Quality (PLREQ'10)

Anil Kumar Thurimella[1], Klaus Schmid[2], Dirk Janzen[3]

[1] Harman International, Boecker-Goering-Str. 16, 76307 Karlsbad, Germany
AnilThurimella@gmail.com

[2] University of Hildesheim, Hildesheim, Germany
schmid@sse.uni-hildesheim.de

[3] metadoc GmbH, Birkenfeld, Germany
di.janzen@web.de

1. Summary of Presentations

The First International Workshop on Product Line Requirements Engineering and Quality (PLREQ'10) was conducted as a half-day workshop. It included two invited talks and two full papers covering various aspects of quality. 17 people attended the workshop and contributed to a lively discussion.

Lauenroth and Heuer presented a tool to deal with the large complexity that product line models can exhibit [1]. They address the complexity by a layering approach in combination with hierarchical models, views, and filters. In particular, they presented a mountain view and a tree view as structuring approaches.

Hubaux et al. [2] show a modeling and reasoning framework for the product instantiation process. They combine formal semantics for feature diagrams [3] and the YAWL workflow language [4] and propose an approach for product instantiation.

Decision-making is an important aspect of product line requirements engineering. Kabbedijk et al. [5] present an empirical study for the determination of characteristics that influence decisions in change management process of requirements engineering. Qualities such as lead time and number of products affected influence decision-making in a market-driven business.

Building new information systems based on a SPL is often less efficient than expected because customer requirements often do not fit the capabilities or constraints of a reuse asset base. Especially in cases where SPL requirements are documented insufficiently, this problem is almost unsolvable. To cope with this challenge, Adam et al. [6] presented a novel approach for flexibly aligning customer requirements with

the capabilities of a SPL. They tailor requirements processes based on the characteristics of a given reuse asset base without the need to reengineer all SPL requirements explicitly.

2. Discussion Points

The discussion focused around a few themes:

Flexibility aspects. Flexibility is a very important quality in product lines for supporting the realization of different products. The key approach to achieving flexibility in product lines is by explicit variability. This flexibility comes with a price: performing changes in a product line platform is more complicated than for a single product and requires a long-term vision.

Modeling quality requirements. While most variability research in product line engineering focuses on functional aspects, it was proposed that more attention should be paid to addressing variability of quality requirements. For example, quality requirements could be modeled in a separate dimension and could be linked to variability models.

Reusing quality requirements. Experience from the automotive sector shows that reusing quality requirements can lead to many errors. The reuse of quality requirements has thus to be done with very much care. Approaches dealing with the reuse of quality requirements and tradeoffs for reuse should be developed.

Domain-specific guidelines. Qualities are handled differently in different domains. Therefore, domain-specific guideline was one more research theme identified in the discussion.

Qualities for benefits. It was also proposed that a more stringent definition of quality benefits in product lines should be developed (e.g., reuse, product qualities, time-to-market) and metrics are needed that allow to quantify these benefits.

References

1. Lauenroth, K. and Heuer, A.: Invited Talk: Tool Support for Model-based Product Line Requirements Engineering - Challenges and Solution Ideas, PLREQ'10, 2010. To be published in this volume.
2. Hubaux, A. Abbasi, E. Classen, A. and Heymans, P.: Workflow-driven Product Derivation, PLREQ'10, 2010. In this volume.
3. Hubaux, A. Classen, A. Heymans, P.: Formal modelling of feature configuration workflow. In: SPLC'09, San Francisco, CA, USA (2009).

4. van der Aalst, W., ter Hofstede, A.: Yawl: yet another workflow language. Information Systems, 30(4) (2005) 245–275.
5. Kabbedijk, J. Wnuk, K. Regnell, B. and Brinkkemper, S.: What Decision Characteristics Inuence Decision Making in Market-Driven Large-Scale Software Product Line Development? In this volume.
6. Adam, S. Doerr, J. Ehresmann, M. and Wenzel, P. Incorporating SPL Knowledge into a Requirements Process for Information Systems – An Architecture driven Tailoring Approach**.** In this volume.

5 First Workshop on Requirements Prioritization for customer-oriented Software-Development (RePriCo)

Editors

Georg Herzwurm
University of Stuttgart, herzwurm@wi.uni-stuttgart.de

Wolfram Pietsch
Aachen University of Applied Sciences, pietsch@fh-aachen.de

Technical Programme

First Workshop on Requirements Prioritization for customer-oriented Software-Development (RePriCo'10) An Introduction

Georg Herzwurm[1], Wolfram Pietsch[2]

[1] Department for Business Administration and Information Systems, esp. Business Software, University of Stuttgart, Keplerstr. 17, 70174 Stuttgart, Germany
herzwurm@wi.uni-stuttgart.de
[2] Business Management, International Sales and Service Management
Aachen University of Applied Sciences, Eupener Str. 70, 52066 Aachen, Germany
pietsch@fh-aachen.de

1 Conception

RePriCo'10 represents the First Workshop on Requirements Prioritization for customer-oriented Software-Development (RePriCo'10) held in conjunction with the 16th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ2010).

Prioritization is an essential task within the requirements engineering process in order to cope with complexity and to establish focus properly.

From a formal standpoint of view it is merely a matter of choice of the right specification method and granularity of analysis. From a practical perspective it is a matter of customer orientation also: consensus must be achieved about the appropriateness of requirements from the view of the customers and fed back into the process.

The workshop served as a platform for the presentation and discussion of new and innovative approaches to prioritization issues for requirements engineering with a focus on customer-orientation. This year's topics dealt with established research findings as well as preliminary research results leading to arguments and positions to be discussed within the community.

RePriCo'10 attracted 11 submissions. Each submission was reviewed by two members of the program committee. Based on the reviews, two submissions were accepted as full research papers and four submissions as position papers.

The research papers comprise current research findings from the fields of requirements elicitation in agile software development and of the support for structured capturing and analysis of requirements from an IT and methodological point of view.

The position papers discuss the customer-orientation of agile software-development methods with respect to requirements prioritization, the need for regarding service commitments in requirements engineering, business value as a

multi-dimensional construct and the combination of prioritization issues with business process-driven requirements engineering.

We are convinced that the findings in these proceedings encourage researchers as well as software-developers, requirements engineers or consultants to absorb new ideas and to carry them out into their daily work and research projects.

2 Organization

2.1 Program Committee

Chair

Prof. Dr. Georg Herzwurm, University of Stuttgart, Germany
Prof. Dr. Wolfram Pietsch, University of Applied Sciences Aachen, Germany

Member

Dipl.-Math. Peter Brandenburg, Vodafone D2 GmbH, Germany
Dr. sci. Math. Thomas Fehlmann, Euro Project Office AG, Switzerland
Prof. Dr. Thomas Lager, Grenoble Ecole de Management, France
Dipl.-Betriebswirt (FH) Olaf Mackert, SAP AG, Germany
Dipl. Wirt.-Ing. Waldemar Meinzer, Volkswagen AG, Germany
Dipl.-Wirt.-Inf. Sixten Schockert, University of Stuttgart, Germany
Prof. Dr. Hisakazu Shindo, University of Yamanashi, Japan
Dipl.-Ing. Gerd Streckfuß, iqm Institut für Qualitätsmanagement, Germany
Prof. Dr. Yoshimichi Watanabe, University of Yamanashi, Japan

2.2 Organizing Committee

Dipl.-Kfm. (FH) Benedikt Krams, University of Stuttgart, Germany
Dipl.-Wirt.-Inf. Sixten Schockert, University of Stuttgart, Germany

Requirements Elicitation in Agile Software Development

Dr. Thomas Fehlmann

Euro Project Office AG, Zeltweg 50, 8032 Zurich
thomas.fehlmann@e-p-o.com

Abstract. Agile development methodologies cope with the problem that requirements are often not detailed or not complete enough to specify all software features needed already in the beginning and in sufficient detail. Adding features step-by-step while developing software to stakeholders' needs is a promising approach. However, how can such software development keep focus on the business goals of the product sponsor? The key is measurement of the Software Process. A combination of Functional Size Measurement and Quality Function Deployment makes the agile development process measurable.

1 Introduction

The Quality Function Deployment (QFD) method orientates products and services towards measurable customer values. It translates between different views of customers and engineers, of users and administrators, or developers and testers. Such translations[1] are called *'Transfer Functions'* **T**; they transfer solutions $\underline{\mathbf{x}}$ into results $\mathbf{T}(\underline{\mathbf{x}})$, called *'response'* [7]. These responses can be measured and compared to the goals $\underline{\mathbf{y}}$. If the variation between goal $\underline{\mathbf{y}}$ and the responses $\mathbf{T}(\underline{\mathbf{x}})$ is within the tolerance range, we understand that these solutions deliver responses according expectations.

The key term is prediction. A prediction model for measurable business goals (the '$\underline{\mathbf{y}}$'s, e.g., defect density, see [12]; project costing[2], see [13]) allows to control the solution-influencing factors '$\underline{\mathbf{x}}$' forcing the response (i.e., the process results $\mathbf{T}(\underline{\mathbf{x}})$) into the allowable tolerance range around the $\underline{\mathbf{y}}$'s.

The entities $\underline{\mathbf{x}}$ and $\underline{\mathbf{y}}$ are vector profiles since neither goal nor influencing factors come alone, or without constraints. The vector spaces that we consider represent business requirements and solution approaches, respectively.

[1] This notion is inherited from the Six Sigma Management initiative, see [7].

[2] Note that we do not use FSM for predictions of effort or duration of a sprint. Experiences have shown that effort prediction based on FSM alone is not very useful for agile development [17]. Effort prediction in agile projects is discussed in [15]; and effort prediction with Six Sigma methods is explained in [13].

2 The Agile Development Cycle

The Agile Development Cycle used in the following sections refers to Scrum, a popular and well-structured agile development methodology [15] whose processes fit well into Six Sigma[3]. The artifacts in Scrum are the *Product Backlog*, the *Sprint Backlog*, and the *Burndown Graph*.

2.1 Planning the Development

The Product Owner builds the Product Backlog, a prioritized list of the client's needs. These needs are expressed and/or written in the form of User Stories [16]. This list is prioritized according to the client necessities; the more business value, the higher on the list the User Story will be placed [18].

Before starting a Sprint, the Scrum Team, the Scrum Master and the Product Owner perform the planning in a ceremony called the Sprint Planning Meeting. During this meeting, the stories with higher priorities are thoroughly discussed and understood by all involved in the process. The Scrum Team then takes these stories and breaks them down into Work Items, representing the activities needed in order to get each story done[4]. These Work Items constitute the Sprint Backlog.

2.2 Planning Sprints

The decomposition of User Stories into Work Items relies on the developers' ability to understand the product sponsor's business. The main problem when doing this kind of decomposition is that the sponsor can only rely on trust into the professional skills and the business domain expertise of developers when judging whether proposed Work Items are really needed and contribute to business goals, and for sorting out gold–plating and efforts waste.

There is no easy work-around for this problem. The literature on Agile stresses the responsibility of the sponsor to decide what is needed for his business and what isn't; however, this is circular reasoning. It is often difficult for the sponsor to decide about usefulness of technical features; developers might have the better judgment. But the developers expect the sponsor to decide; whereas the sponsor believes the developers should know better what is needed.

In Scrum, it is left to the Scrum master to moderate between product sponsor and team which feature to implement and which not. This procedure is not inspiring innovation and search for inventive solutions. Because of this vicious cycle, any software development inherently carries the risk of loosing focus on business needs.

---

[3] This is for convenience only; in fact, the Six Sigma techniques presented here can be used for all agile development approaches that derive from the ideas presented originally by Kent Beck [14].

[4] We prefer using the term "*Work Item*" instead of activities, since performing activities do not necessarily add value.

Involving the customer, represented by the product sponsor, into design decisions does not address that risk.

And even if the product sponsor is technically skilled enough to prioritize Work Items, loading technical decision responsibility upon the sponsor makes poor use of the advantages of division of work according professional skills. It is not normally expected that the product sponsor is equally skilled in ICT technology as the developers are.

The usual work-around for this problem is remaining high-level when prioritizing Work Items for a Sprint. Another work-around is hiding; creating clandestine Work Items when technically needed but impossible to explain to the product sponsor. Both tactics are used frequently in software development – not only in Agile! – and constitute serious threats for business success.

2.3 Validation and Entering Next Cycle

At the end of the Sprint an increment of the product is delivered. The updated Burndown Graph visualized the progress made. Two ceremonies are performed, the Sprint Review, where the potentially deliverable product increment shall be accepted by the Product Owner, and the Sprint Retrospective, where the team discusses improvements that can be done during the next Sprint. These improvements range from changes in artifacts to better personal interaction.

The development project closes when the Product Backlog is empty; or the Sponsor wants to move the product into maintenance phase; or development focus changes from new features to feature enhancements and bug fix requests that might have piled up.

2.4 The Need for Measurements

The stakeholders in agile software development need software metrics that are similar to traditional software development. They need measurements for:

- Development effort – is there some budget left?
- Size metrics – how big is the backlog?
- Quality metrics – how satisfied users and customers will be with the product?

Development effort is measured by time-boxing. For sizing, *Story Points* are a popular method for sizing backlogs, or requirements. However, Story Points is not a measurement method. It lacks standardization, repeatability and independence from teams [16]. For quality, often enough only qualitative criteria exist.

Measuring software already during the software development cycle in turn has no disadvantages. It hurts nobody and nothing, especially not in an environment where people are empowered to perform agile software development.

3 Functional Size Measurement Methods

There are many functional size measurement methods; the most popular being the Function Points counting model according IFPUG (ISO/IEC 20926) [2] and the COSMIC measurement method (ISO/IEC 19761) [4]. Both methods measure different aspects of software requirements [11], and thus it depends on the business environment which method to choose. For this paper, we concentrate on the COSMIC method.

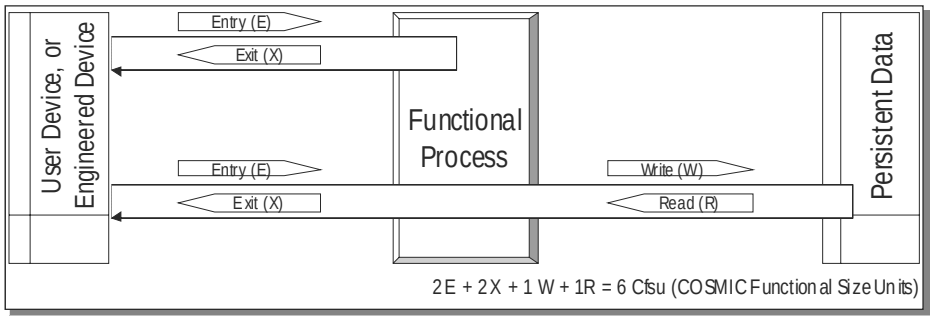
3.1 COSMIC Functional Size

For the COSMIC measurement method it matters how many interactions are needed between system components to produce a system response.

3.1.1 Identify Data Movements

An occurrence of a data movement is a base functional component which moves a single data group type. There are four sub-types of data movement types: Entry, Exit, Read and Write.

Fig. 1: Visualization of Counting Data Movements



A data movement occurs during the execution of a functional process. There are four sub-types of data movement: Entry, Exit, Read and Write, each of which includes specific associated data manipulation.

1. An Entry (E) is a data movement that moves a data group from a functional user across the boundary into the functional process where it is required. An Entry is considered to include certain associated data manipulations.
2. An Exit (X) is a data movement that moves a data group from a functional process across the boundary to the functional user that requires it. An Exit is also considered to include certain associated data manipulations (e.g. formatting and routing associated with the data to be exited).
3. A Read (R) data movement that moves a data group from persistent storage within reach of the functional process which requires it. It is considered to include certain associated data manipulation sub-processes necessary to achieve the Read.

4. A Write (W) is a data movement that moves a data group lying inside a functional process to persistent storage. It is considered to include certain associated data manipulation sub-processes necessary to achieve the Write.

Data movements can be derived from the components identified in sequence diagrams according the RUP Methodology, see [19], [20].

3.1.2 Counting Data Movements

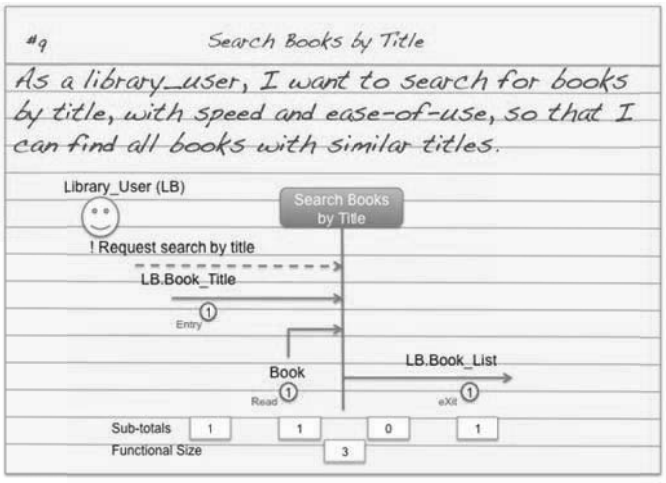
Counting COSMIC functional sizing units is simple: you sum up all the data movements identified above.

COSMIC allows accounting for the complexity of distributed data storage due under appropriate viewpoints, e.g., some particular architecture, as this may cause large number of Writes in COSMIC, but yielding only one Elementary Process. The COSMIC counting method is preferred for modern architectures and for real-time applications.

3.1.3 Visualization

Visualization is also available, see Fig. 1. This counting method fits very well to User Story cards, as pointed out by Grant Rule in a recent note (Fig. 2)[5]. It creates a common understanding and allows identifying risks with the planned software at an early stage.

Fig. 2: Sample User Story with FSM according ISO/IEC 19761[6]



3.1.4 Identifying Action Items

From Fig. 2, we identify three data movements as Work Items needed to implement the User Story “Search Book by Titles”: “Read Book Title”, “Search in Book Data”,

[5] Communication by Charles Symons, chairman of COSMIC Consortium (www.cosmicon.com)

[6] Sample User Story card with acknowledgment to the creators Grant Rule, Charles Symons (both Software Measurement Services Ltd) and Peter Fagg (Pentad Ltd.).

"Present list Books Found"; an Entry, a Read, and an eXit. These are functional tasks that the software must be able to deliver in order to implement the said User Story. But these three Work Items are not enough.

3.2 Extension of Functional Requirements to Work Items

The value of the sequence diagram to detect software risks and missing requirements is known and well-documented [19]. The strict definitions used for defining elementary processes and data movements in the international standards provide a means to detect missing requirements before development has started.

Note that are four more Work Items hidden in the user story in Fig. 2 that are needed to implement the User Story to the customer's full satisfaction:

1. the need for an extended title text search mechanism;
2. search speed optimization;
3. an entry scan that forgives grammar glitches; and
4. a pattern matching search algorithm.

Although the FSM method serves well for decomposing User Stories into measurable and executable Work Items, it does not identify non-functional requirements. The notions of elementary processes in FMS such as data transactions, resp. data groups, are far away from actual data storage, from architectural constraints, or from performance concerns. For instance, in our sample User Story in Fig. 2, we would need additional environmental information that probably only developers have.

Developers have to do things like setting up one or more databases or services, in other – more traditional – words: they have to design a solution, before they can implement elementary processes. FSM simply does not give all requirements for the design, because it is limited to user-perceivable functionality. Neither technical nor quality requirements are detected by FSM.

4 Measuring non-functional requirements

4.1 Quality Function Deployment

Quality Function Deployment (QFD) has come to life in Japan during the 1980ies and had been applied in the heavy industry first for determining customer's needs, not in terms of functionality alone but also in quality [8]. Today, QFD is widely used in the automotive industry, but also in software ergonomics, marketing services, and decision making. The mathematical foundation of QFD is the Eigenvector theory, similar to what Google uses for its search engine [24].

The basic idea behind QFD is to translate the profiles of goal topic into profiles of a suitable solution topic by means of a linear transformation. A profile is a vector in a multi-dimensional vector space, usually in the space of independent events that can happen. In a traditional industrial production process, events are physical; in software, statistical control of events happens in the space of mutually independent requirements. Requirements are less prone to measurements than events in a

production process, and orthogonality is even more disputable. This makes QFD more difficult to apply for software; however, the mathematics for QFD is the same.

The term "Critical Parameter" is used for setting up the profile vector. Criticality implies Orthogonality, see [7], pp. 87. The Transfer Function achieves the linear mapping of the Critical Parameter Profile to Critical Functional Responses as a mathematical transformation. The identification of profiles in specific product-related topics is called "Critical Parameter Management" (CPM), see [7], pp. 130, 259-261, 280ff etc.

The *goal profile* in Six Sigma, the defect-free ideal result of the process, is called $\underline{\mathbf{y}}$. If $\mathbf{T}$ is the known Transfer Function, the challenge is to find the *solution profile* $\underline{\mathbf{x}}$ such that $\mathbf{T}(\underline{\mathbf{x}}) = \underline{\mathbf{y}}$. This means, we are looking for the process input parameter profile $\underline{\mathbf{x}}$ such that $\mathbf{T}(\underline{\mathbf{x}})$ yields $\underline{\mathbf{y}}$ as exactly as possible. Finding the solution profile $\underline{\mathbf{x}}$ is not that easy. We would need the inverse Transfer Function $\mathbf{T}^{-1}$ such that we can effectively compute $\underline{\mathbf{x}} = \mathbf{T}^{-1}(\underline{\mathbf{y}})$. In general, such an inverse does not exist[7].

Fortunately, we don't need to compute the inverse Transfer Function for a solution; we only need a good guess for $\underline{\mathbf{x}}$ yielding a $\mathbf{T}(\underline{\mathbf{x}})$ near enough to $\underline{\mathbf{y}}$. 'Near' means the usual vector distance: let $\underline{\mathbf{y}} = \langle\psi_1,\ldots,\psi_n\rangle$ and $\underline{\mathbf{z}} = \langle\zeta_1,\ldots,\zeta_n\rangle$ be two vectors of dimension *n* in the goal space, then the vector distance is the square root of the sum of squares. For normalization reasons, we divide[8] the result by square root of the dimension *n*:

$$||\underline{\mathbf{y}} - \underline{\mathbf{z}}|| = \sqrt{\frac{\sum_{i=1..n}(\psi_i - \zeta_i)^2}{n}} \qquad (\mathbf{1})$$

4.2 The Convergence Gap

With QFD, people search for solutions among vector profiles of the form $\mathbf{T}^T(\underline{\mathbf{y}})$. If $\underline{\mathbf{x}}$ is a solution, i.e., $\underline{\mathbf{x}} = \mathbf{T}^T(\underline{\mathbf{y}})$ for some Transfer function $\mathbf{T}$, and $\mathbf{T}(\underline{\mathbf{x}})$ comes near to $\underline{\mathbf{y}}$, i.e. $||\underline{\mathbf{y}} - \mathbf{T}(\underline{\mathbf{x}})||$ is small, then the solution $\underline{\mathbf{x}}$ is a considered as a possible problem solution. Otherwise, if the solution profile $\underline{\mathbf{x}}$ delivers a $\mathbf{T}(\underline{\mathbf{x}})$ that is far away from the goal profile $\underline{\mathbf{y}}$, the Transfer function $\mathbf{T}$ yields on $\underline{\mathbf{x}}$ something else than the customer wanted.

If the goal profile $\underline{\mathbf{y}}$ is of dimension *m*, the vector distance is measured in the goal topic space with dimension *m* again. This distance is a metric for how accurate the solution profile $\underline{\mathbf{x}} = \mathbf{T}^T(\underline{\mathbf{y}})$ solves the problem $\underline{\mathbf{y}} = \mathbf{T}(\underline{\mathbf{x}})$[9]. We can compute the distance between goal and response, using formula **(1)**:

---

[7] If the inverse Transfer Function would always exist, we would have a general solution solver for all kind of problems. Such a miracle solver would make business life boring.

[8] The reason for the division through the square root is discussed in full in [9].

[9] Six Sigma metrics are different from other metric approaches: metrics are in the goal topic area, not in the solution area. Traditional metrics most often focus on solution topics, e.g., bug counts in software. Six Sigma is not looking for solution-approach metrics. Two solution approaches that yield the same responses are equivalent for Six Sigma metrics.

$$| | \underline{\mathbf{y}} - \mathbf{T}(\underline{\mathbf{x}}) | | \qquad (2)$$

This is called the *Convergence Gap* for the solution $\underline{\mathbf{x}} = \mathbf{T}^{T}(\underline{\mathbf{y}})$. Formula **(1)** describes the Convergence Gap in component form when replacing $\underline{\mathbf{z}}$ with $\mathbf{T}(\underline{\mathbf{x}})$. From a practical point of view, using $\underline{\mathbf{x}}$ as process input, the Transfer function **T** enforces the response $\underline{\mathbf{y}}$.

4.3 The Business Value of QFD

The agile development processes, as exemplified with Scrum, already has a mechanism to define **T**. When the Scrum team defines the Work Items needed, it implicitly finds the solution profile $\underline{\mathbf{x}}$. All that is left is checking the Convergence Gap to see whether $\underline{\mathbf{x}}$ will deliver what the sponsor expects. The $\underline{\mathbf{y}}$'s are the customer's business values; the $\underline{\mathbf{x}}$'s are the quality criteria that the project team must meet to deliver the expected business values. This is the essence of the QFD process [23].

4.4 Blending Quality Function Deployment with Agile

There is a simple and well-established visualization technique available that was developed at the Kobe shipyards in the 1980's.

The $\underline{\mathbf{y}}$-axis of our matrix is labeled with the business requirements. These requirements must not be functional but describe what qualities matter for the customer's business. Speed, performance, safety, security, privacy, relevance, correctness, completeness are typical for such values that derive from the customer's business. They can be specific like 'personal health data of employees is accessible to Human Resource People only', or describe soft values like 'Create Trust among Users'. Business requirements should not prescribe technical solutions (like 'must be implemented with MySQL'); business requirements must be something a developer can have impact upon, rather than solution constraints.

Similar to the FSM approach, a team can visualize a QFD matrix without bothering about numbers and theory. The $\underline{\mathbf{x}}$-axis is labeled with the User Stories that the developers want to implement. The entries in the correlation matrix cells represent the Work Items needed to implement a User Story, identifying the respective business requirement where it contributes. The rules for the numerical matrix evaluation are given above and are not more complicated than the Low-Medium-High characterization of complexity of elementary processes. Numerical evaluation can be done off-line and probably has less relevance for the developers than for the Product Sponsor.

The rows ($\underline{\mathbf{y}}$-axis) are labeled with the goal topics: the (non-functional) business requirements; the columns ($\underline{\mathbf{x}}$-axis) contain the solution topics: the User Stories. The *correlation values* ($\mathbf{a}_{i,j}$) – the matrix components of the Transfer Function **T** – are found as follows: in each sprint, the developers identify the Work Items needed for an User Story. Each such Work Item can be entered into one or more matrix cells if it contributes to the respective business requirement. Work items are not restricted to functional features; typically they refer to work enhancing quality rather than

functionality, such as creating a database schema that is extendible in future version of the software, re-factoring a feature for better performance, adding features for better ease-of-use; we don't restrict the creativity of developers to argue why they need to perform such Work Items. Only, they must contribute to some business requirement.

4.5 Collecting Developer's Intelligence with QFD

The evaluation of the matrix cell is done as follows: the developers that worked on some user story have points to distribute to all Work Items recorded in the matrix cells according the functional size of the respective User Story. They can distribute those points to the Work Items, indicating how much they did contribute to each of the business requirements.

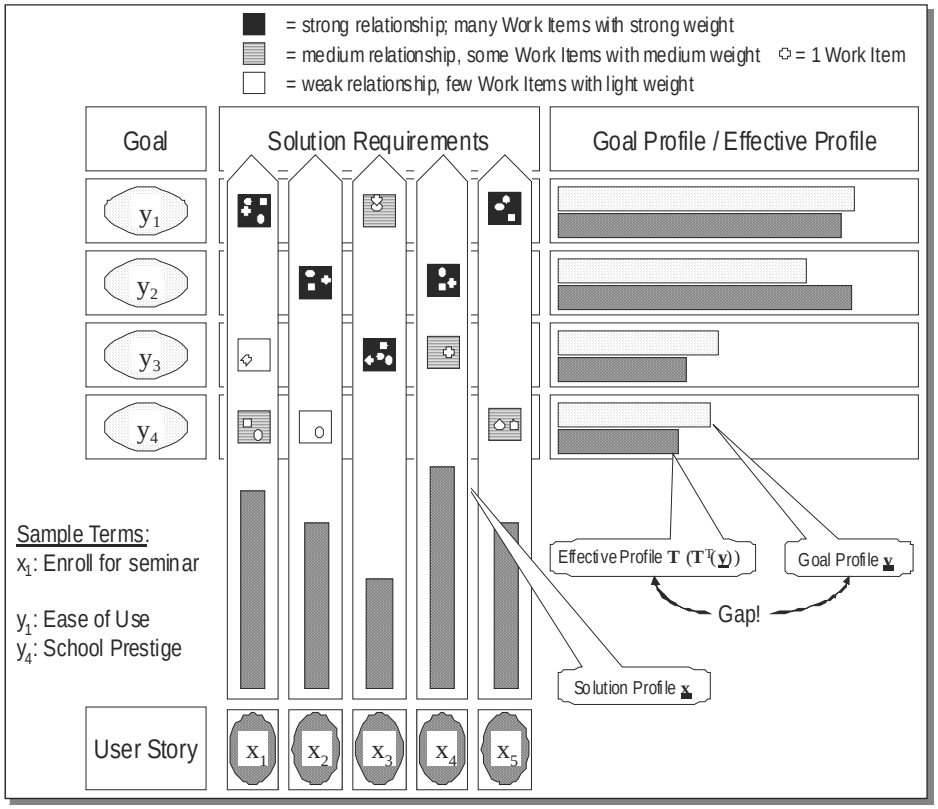
This can be done initially, at Planning Poker time, or after the daily stand-up meetings (see section 4.7 Focus on Quality thanks to Developer's Empowerment), and revised whenever needed. The developers use colored marker buttons that they can distribute; red points represent value 6, blue points represent value 3, and light green points represent value 1. The total of points distributed per User Story equals its FSM count.

Note that one Work Item may appear several times in a column, if it contributes to more than one single business goal. The total weight attributed to all Work Items in a column must equal its total functional size.

The total value of a correlation cell $\mathbf{a}_{i,j}$ is the sum of all points for the Work Items recorded in the matrix cell[10] that contribute to the respective User Story. We use colored "Post-it®" sticky notes per Work Item identifying the Work Item by its number.

[10] Note that the same feature can be used in different cell if that Work Item contributes to more than one business requirement. However, the total number of value points is limited by the functional size of the respective User Story.

Fig. 3. The QFD House of Qualities[11]



The result is a visualization that is immediately understood by developers and customers alike. Areas with strong correlation in the matrix contain much red, indicating important solution topics for reaching the stated goal. Areas that contribute less remain lighter colored or even white[12].

People visually perceive a matrix with large Convergence Gap as unbalanced; and well-balanced matrices in turn have small Convergence Gaps. Because of this, QFD has been very successfully applied as a workshop technique in many industries without people ever understood the Eigenvector theory behind [21], [24]. In our context, we use QFD as a communication tool, not for its statistical significance alone.

[11] Following a remark by Glenn Mazur, both matrix dimensions address qualities: one the business goals, the other the solution qualities of the Work Items.

[12] In QFD, finding the matrix describing the Transfer Function is the problem. To identify the correct correlation values, you need excellent data. Google finds the data using a Web Crawler that correlates web pages with its content. If you have no data to fill in the matrix elements, you try best guesses from QFD workshops [8] with excellent people. In agile software development, the developers fill the correlation cells in the House of Qualities matrix, using their technical skills and business intelligence.

Finding all Work Items with functional decomposition using the FSM method alone is not possible. The case with the database setup was already mentioned, but more such features come easily to mind. The QFD policy enables developers to add non-functional Work Items to user stories when there is a fitting matrix cell. Using this policy, developers have a standardized, institutionalized way to assess usefulness of Work Items that they expect contributing to quality. For this reason, the correlation matrix is called *House of Qualities*.

This correlation matrix is constructed in a different way from traditional QFD. Traditionally, the correlation elements are found by physical evidence rather than by identifying qualities in people's Work Items.

We expect the completed House of Qualities to be balanced in the sense that all requirements are addressed but none is overachieved, as seen with the horizontal bars in Fig. 3. This means the team spent Work Items on requirements according importance for the customer. Mathematically speaking, its Convergence Gap is low. The business understanding is that a low Convergence Gap indicates that all of the business requirements are addressed, and none over-achieved. Using the House of Qualities, the Scrum master can visualize and explain it to both product sponsor and the team.

4.6 Using QFD for User Story prioritization

The traditional use of the QFD matrix – not only in Agile – is for prioritizing User Stories. In fact, when one knows the necessary Work Items in advance, summing up their contributions and weighting it with the goal profile $\mathbf{y}$ is the traditional method of identifying the initial solution profile $\mathbf{x}$. In our context, this is also possible. A User Story that does not contribute to business goals at all is probably not needed. However, the primary goal of our use of the QFD matrix is identifying and allocating Work Items to User Stories, the correlation elements in that matrix are typically not yet known when selecting User Stories for implementation.

4.7 Focus on Quality thanks to Developer's Empowerment

Additionally to the functional decomposition with FSM, each developer who identifies a non-functional or functional requirement as a Work Item to the Sprint backlog will be asked to position his Work Item in the House of Qualities in the respective User Story column. The developer can distribute as many correlation weight points to the matrix cells in the House of Qualities as the User Story had in total for functional size. Thus the total strength of correlation is weighted by functionality; the reason for this is that quality can only be perceived through functionality. It is the developer's decision whether to spend all his functional weight in one cell, or spreading them more or less equally according the emphasis that had been set on the particular customer requirement.

The House of Qualities grows over time with each Work Item identified as a requirement, and the Convergence Gap gives an accurate measurement whether the product development will meet the quality goals of the customer.

5 Conclusion

5.1 Requirements Elicitation with Agile

Agile software development has big advantages over traditional approaches in detecting missing requirements early enough, making it worthwhile becoming measurable and traceable for management. It encourages developers to bring in their experience for requirements elicitation throughout the development phase. Thanks to the control options offered by measuring functional size and the quality deployment convergence gap, agile development becomes eligible for mission-critical software.

5.2 The Need for a Project Office

A Project Office is needed that can deliver the Six Sigma methods and tools in software development and support the project manager, Scrum master and product sponsor effectively. The Project Office's task are mastery of tools and methods, set up measurements, and providing the stakeholders including Scrum Master and Product Sponsor with the necessary metrics, analysis and reports. Project Office services need being included in a project's budget if the Six Sigma approach is going to work for software development.

5.3 The Upcoming Paradigm Change in Software Economics

The software industry is the main motor for economic growth in the 21st century. Software development and integration of ICT services deliver business value – if the delivery is in time and meets business needs. The combination of agile development methods and Six Sigma tools and techniques is a paradigm change for requirements elicitation.

6 References

1. Albrecht, A.J.: Measuring Application Development Productivity. In: GUIDE/SHARE: Proceedings of the IBM Applications Development Symposium, Monterey, CA (1979).
2. International Function Point Users Group (IFPUG): Function Point Counting Practices Manual, Release 4.3, Princeton Junction, NJ (2010)
3. Abran, A,. Robillard, P.N.: Identification of the structural weakness of Function Points metrics. In: 3rd Annual Workshop on Software Metrics, Portland, Oregon (1991)
4. Abran, A. et. al.: The COSMIC Functional Size Measurement Method – Version 3.0.1 – Measurement Manual (2009), http://www.cosmicon.com/
5. Lokan, Ch.J.: Function Points. In: Advances in Computers, M. Zelkowitz (ed.), Volume 65, Chapter 7. Academic Press (2005)
6. CMMI® for Development, Version 1.2, Carnegie-Mellon University, Software Engineering Institute, Pittsburgh, PA, (2006)

7. Creveling, C.M., Slutsky, J.L., Antis, D.: Design for Six Sigma, Prentice Hall, New Jersey (2003)
8. Herzwurm, G., Schockert, S.: What are the Best Practices of QFD? In: Transactions from the 12th International Symposium on Quality Function Deployment, Tokyo, Japan (2006)
9. Fehlmann, Th.: The Impact of Linear Algebra on QFD. In: International Journal of Quality & Reliability Management, Vol. 21 No. 9, pp. 83--96, Emerald, Bradford, UK (2004)
10. Fehlmann, Th.: Statistical Process Control for Software Development – Six Sigma for Software revisited. In: EuroSPI^2 2006 Industrial Proceedings, University of Joensuu, Joensuu, Finland (2006)
11. Fehlmann, Th.: When use COSMIC FFP? When use IFPUG FPA? – A Six Sigma View. In: MetriKon 2006 – Praxis der Software-Messung, Tagungsband, Potsdam, Germany (2006)
12. Fehlmann, Th.: Defect Density Prediction with Six Sigma. In: Proceedings of the 6th Software Measurement European Forum, Rome, Italy (2009)
13. Fehlmann, Th.: Using Six Sigma for Software Project Estimations. In: MetriKon 2009 – Praxis der Software-Messung, Tagungsband, Kaiserslautern, Germany (2009)
14. Beck, K.: extreme programming explained, Addison-Wesley, Boston (2000)
15. Schwaber, K., Beedle, M.: Agile Software Development with Scrum, Prentice Hall (2008)
16. Cohn, M.: Agile estimating and planning, Prentice Hall, New Jersey (2005)
17. Fuqua, A.M.: Using Function Points in XP – Considerations. In: Extreme Programming and Agile Processes in Software Engineering, Lecture Notes in Computer Science, pp. 1013ff
18. Heitor Roriz Filho: Can Scrum Support Six Sigma? In: Scrum Alliance (2009), http://www.scrumalliance.org/articles/161-can-scrum-support-six-sigma
19. Bell, D.: UML basics: The sequence diagram – introductory level. In: IBM Developer Works (2009), http://www.ibm.com/developerworks/rational/library/3101.html.
20. OMG Unified Modeling Language™ (OMG UML), Superstructure V2.2 pp. 506--524 (2009), http://www.omg.org/spec/UML/2.2/Superstructure/PDF/
21. Saaty, Th.L.: Decision-making with the AHP: Why is the principal eigenvector necessary. In: European Journal of Operational Research vol. 145, pp. 85--91, Elsevier Science B.V. (2003)
22. Saaty, Th.L., Ozdemir, M.: Negative Priorities in the Analytic Hierarchy Process. In: Mathematical and Computer Modeling vol. 37, pp. 1063--1075, Elsevier Science B.V. (2003)
23. Johnson, C.M., Mazur, G.: Value Based Product Development – Using QFD and AHP to Identify, Prioritize, and Align Key Customer Needs and Business Goals. In: Transactions from the 20th Symposium on Quality Function Deployment, Santa Fe, NM (2008)
24. Kressner, D.: Numerical Methods for General and Structured Eigenvalue Problems. Springer, Heidelberg. Lecture Notes in Computational Science and Engineering, vol. 46 (2005)

Data Processing System for structured Capturing and Analyzing of Requirements

Sandra Klute[1], Constanze Kolbe[1], Robert Refflinghaus[2]

[1] Technische Universität Dortmund, Lehrstuhl für Qualitätswesen,
Joseph-von-Fraunhofer Str. 20,
44227 Dortmund, Germany
Constanze.Kolbe@lqw.mb.tu-dortmund.de
[2]RIF e.V. ,
Joseph-von-Fraunhofer Str. 20,
44227 Dortmund, Germany

Abstract. For planning and developing complex products a great amount of stakeholder requirements has to be considered and implemented into solutions. To guarantee adequate requirements management, requirements have to be gathered and structured and interdependency and relations between requirements have to be analyzed. For this purpose a requirements management system has been developed with the German Collaborative Research Centre 696. Therefore, multi-dimensional structuring model has been developed and implemented for data processing. Moreover, the system comprises ontology and a wiki. The purpose of this paper is to present the developed requirements management system and the collaboration of its different components.

Keywords: requirements management, structuring, ontology

1 Introduction

When planning complex products like for example intra-logistical facilities or software a multitude of stakeholders with different requirements has to be taken into account. Although the stakeholders exchange information, there are often problems regarding the meaning and content of the requirements. These problems result from the stakeholders' different professional and functional background. Beyond that, it has to be considered, that the requirements may be very different from each other or even conflictive due to the different aims of the stakeholders. Furthermore, an adequate and systematic requirements management is difficult because of the great amount of requirements which have to be handled. Combined these aspects entail inconsistency and uncertainty for the planning process. Decisions are partially made on stakeholder requirements, but mostly based on knowledge and experience resulting from former projects.

For an efficient requirements management consequently, structuring and data processing of the gathered requirements is essential. Therefore, on the one hand, a model, which allows structuring all requirements on the regarded product, on the

other hand a system for data processing which provides the information formatted for the planning process is necessary.

However, for data processing a textual description of the requirements is not sufficient. In fact, the requirements have to be highly formalized. For formalizing requirements which is adapted to a special domain ontology can be used [1]. Furthermore, the formalizing should provide an aid for the stakeholders for interpreting and understanding of the requirements. Hereby, conflicting requirements which cannot be transformed together into adequate product characteristics have to be considered. To analyze such relations between requirements a knowledge base is necessary in which knowledge about the interrelationship of the regarded domain is stored formalized. Based on the relations between requirements and the model for structuring requirements it is possible to gain findings for the implementation of the requirements in advance. Hence, these could be taken into account in the planning process, e.g. when applying quality function deployment (QFD). QFD is an established quality management method for transforming requirements into product characteristics [2].

The formalization as well as the further processing of the requirements hinge on the formalized representation of the domain knowledge within the ontology. Due to the fact, that many terms arise out of the course of time or are difficult regarding the requirements' interpretation, special mechanisms for enlarging are necessary. Moreover, it is not sufficient to deal with single requirements; rather a topical structuring of the requirements is necessary for a transfer into QFD and for assuring that there are no information deficits regarding the requirements. Therefore, a multi-dimensional structuring model has been developed which allows structuring all requirements on the reference object. This should optimize the planning process in the long run.

In this paper the requirements management system will be presented. Furthermore, the interrelation between the different components of this system will be shown.

2 Requirements Management System

The developed requirements management system consists of different components which are linked to each other (fig. 1). In the following these components will be presented.

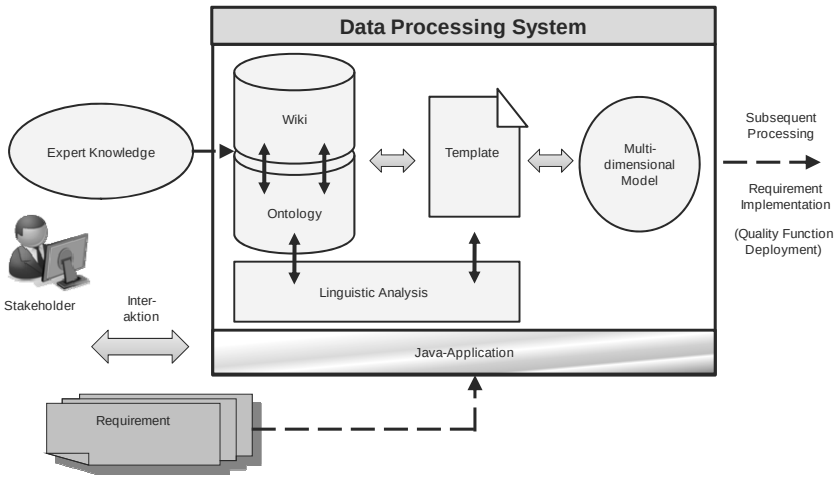


Figure 1: Components of Data Processing System

2.1 Template (Formalization of single requirements)

The intention was to structure single requirements with regard to their content and to document the requirement preferably without loss of information. A model in the term of a "requirement template" was developed in order to formalize single requirements and to make them machine-readable (fig. 2). Therewith, information which represents the content of a requirement can be mapped structured. The different information categories of the template were summarized up to five headlines. These headlines are the "reference", the "source", the "weighting" (for the later prioritization), the „constraints" (for the analysis of relations of requirements) and the field „situation" (e.g. time of capturing for versioning of the requirements amount). The category "reference" defines the core requirement and the other categories contain extended information of the requirement. Requirements can be described by using three components [3]. These components which are summarized under the headline "reference" are the reference object, the attribute and the value. For instance, the requirement "The system should offer a high availability." can be transformed in the triple reference object „system", attribute „availability", value „high" [4]. A variety of scale types can be used to illustrate the value of a requirement – nominal, ordinal, interval, ratio and absolute [5]. The lowest scale level is the nominal scale and the highest scale level is the absolute scale [6]. The higher the scale level used, the higher the volume of information regarding the requirement.

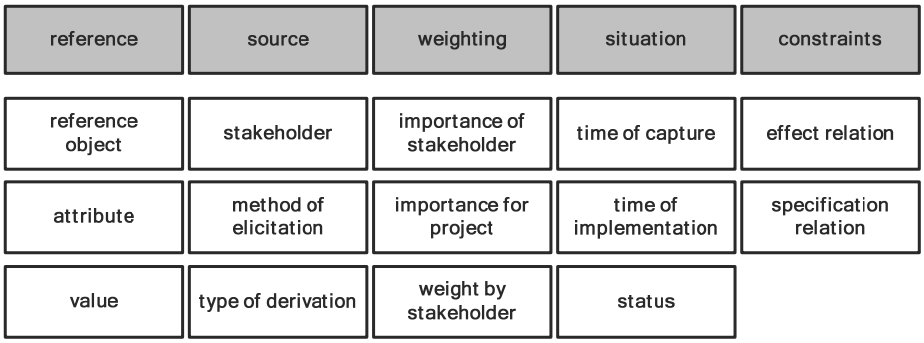


Figure 2: Template for the Depiction of a Requirement

This requirement template is realized as a java-application and was tested by collecting over 200 requirements from the field of intra-logistics. Ranges for each information category were defined to simplify the appliance of the template. For instance, methods of elicitation like qualitative or quantitative methods are selectable by the stakeholder. But this simple kind of pre-structuring of requirement information is not adequate for the information category „reference“ as the system has been developed for a complex product with a high amount of relationships. For this purpose, ontology was developed in order to depict requirements, their three components and their relationships between each other.

2.2 Ontology, Wiki and Linguistic Analysis (Storage of knowledge about single requirements)

After defining the components for the depiction of a requirement by a template a knowledge base for possible requirements and their relationships was established. The knowledge base was developed in terms of ontology by using the editor Protégé. Ontology is “a formal specification of a shared conceptualization” [7]. The developed ontology enables the formalization of only meaningful requirements and allows their comprehension by the computer as well as the identification and establishment of relations between requirements. Therewith, the problem of computer-based comprehension of requirements, that is one of the main issues of requirements management, could be solved. Ontology consists of a class hierarchy which is made up of relevant terms and properties. The properties specify the connection of those classes between each other.

ONTOLOGY (machine readable knowledge)

The class hierarchy of the established ontology is composed of the main classes “requirement sentence“ as well as of classes which represent the trisection of requirement “reference objet“, “attribute“ and “value“ (fig. 3). In the upper class, “requirement sentence“ entities are generated. Each of these entities represents one single requirement.

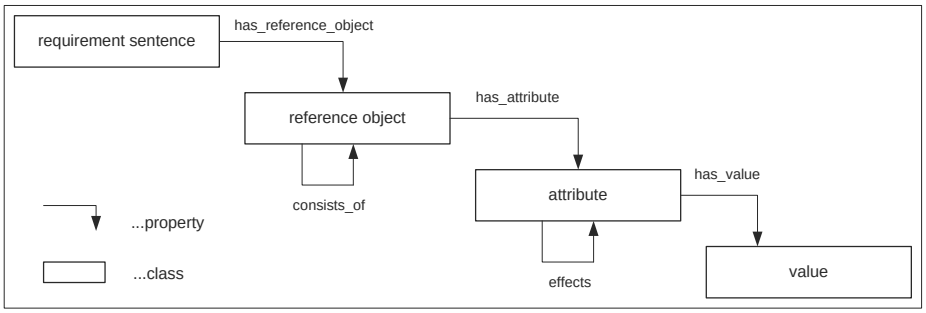


Figure 3: Schematically overview of a requirements-ontology

From this class a “has-reference-object”-property goes to the second upper class “reference object“ (with regards to content is this class equivalent to the category “reference object“ of the multi-dimensional model). The next property, called “has_attribute” is defined between the class “reference object” and the “attribute”-class. And a “has_value”-property points from the class “attribute“ to the fourth upper class “value“. These mentioned properties define only sensible connections between the classes and sub-classes, which means that only meaningful combinations of reference object, attribute and value are determined. These combinations comprise all potential requirements that could occur during the gathering of requirements. By implication, only sensible requirements can be collected by the template. There are also „consists_of“- and „effects“-properties set in the ontology. The “consists_of”-property defines which reference object could potentially consists of another reference object. This kind of property allows describing a product structure during a planning process. Knowledge about the effects and fulfillment of one requirement can have on the fulfillment of another requirement can be stored in the ontology by using “effect“-properties. In summary, the ontology acts as a library of possible requirements and their relations. With every further planning process of a product the knowledge base increases. The library also supports the assurance that all important requirements are formulated by the stakeholders. The probability that some of the important requirements are not stated by the stakeholders can be reduced.

The ontology especially conduces to permit a machine-comprehension of the requirements. But also the stakeholder as a human has to understand the terms located in ontology. Problems of comprehension do also occur between the stakeholders as they belong to different disciplines. Due to that they use different terms for the same meaning or the same term but with different interpretation. For that reason a wiki was constructed that contains the necessary terms and their interpretations.

WIKI (human readable knowledge)

The wiki consists of a system of web-sites ([8], [9]) whose content can be read and also edited by the stakeholders. It provides definitions of the ontology´s terms as well as further definitions from the related fields of intra-logistics, like quality management plus instructions for a correct use of the management system itself. The collected terms and definitions of the wiki are linked in order to describe their relations between each other. The wiki supports the stakeholder during the requirement formalization as well as during the whole planning process because ambiguity and wrong interpretations of requirement can be minimized.

LINGUISTIC ANALYSIS
Because of the high amount of requirements the manual formalization of requirements can get quite elaborate. Due to that, a linguistic analysis (shallow parsing) as a further software-component was integrated to the system. This component enables the automatic computerized formalization of requirements that are mentioned in natural language. The terms that are contained in the requirement text and their reference to one another can be determined by the analysis while matching with ontology. The stakeholder can confirm the result of the analysis or carry out a manual correction of the results.

2.3 Multi-dimensional Model

For the process of structuring the requirements, which have been gathered beforehand, a multi-dimensional model has been developed. This model is a generic approach, although it is been developed for the area of intra-logistics. Therefore, it is generally applicable and extendable.

The model allows structuring the requirements from the stakeholders' point of view. Additionally, the aspect of time had been considered. In this context, it had to be taken into account that requirements are of different importance in different phases of lifecycle. Moreover, requirements are not static but dynamical. That means that they change in recourse of time regarding their importance and their level of specification. Hereby, the level of requirements' specification often is rather low at the beginning of the planning process and rises during the different stages of planning. Moreover, customers may not be able to articulate all of their requirements at the beginning of the planning process [10].

Beyond that, the model is holistic and includes requirements and their fulfillment and the thereby resulting customer satisfaction. This allows feedback between requirements and their implementation.

Because of the multitude of stakeholders and requirements a multi-dimensional structuring model is necessary to cope with the thereby caused great complexity. By choosing the number of structuring dimensions and the division of each dimension into categories a space can be generated, in which requirements can be classified. Hereby, the corresponding dimensions and categories should be chosen in a way that they are associable to the requirements of complex product, in the case at hand an intra-logistical facility. Avoiding laminations and providing independencies of these categories should be taken into account. Classes of different dimensions should therefore be not too similar in order to prevent a comparison of these two categories from having no validity. However, categories, which belong to different dimensions, deal with the same subject, but from a different point of view and often with different focus. For example, the subject "environment" is dealt with in the dimension "obligations" and in the dimension "surroundings". Thereby, the first one is considered with aspects like environmental protection and the latter with environmental aspects which are not necessarily dealt with by legal aspects. Furthermore, classes should not depend on each other. That means that the classification of one requirement is not based upon the results from a requirements structuring of another dimension. [11]

It should also be taken into account, that consequently, it is not possible to classify requirements exactly in one dimension. They should rather be classified in an n-dimensional space, which comprises all dimensions of an intra-logistical facility or rather complex product occurring while planning. Referring to the developed model and with respects to the field of application there are 9 dimensions: obligations, surroundings, economy, information, qualification, technical and functional requirements, product, evaluation respectively weighted level of performance and customer satisfaction. Additionally, the time dimension has to be considered. It may be not an independent or comparable dimension to the other ones, but it should be taken into account, because every requirement always includes a temporal aspect. For example requirements can occur in the phase of planning or operating (figure 4)

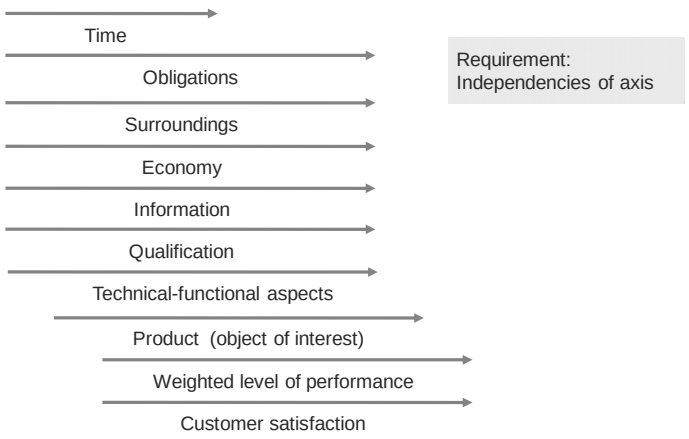


Figure 4: Multi-dimensional Model for Structuring Requirements

Moreover, it should be taken into account that the content and meaning of the dimensions product, weighted level of performance and customer satisfaction are different from the other dimensions. The dimension product serves to structure the reference object to which every requirement is related. The other two dimensions serve to give feedback to the requirements that were gathered and structured previously. These dimensions allow verifying how and to what extent the requirements of the stakeholders are fulfilled. The model's dimensions which are used to gather and structure the requirements are therefore reflected in the dimension weighted level of performance, which represents consequently the actual condition whereas the former ones represent the nominal condition. Comparing actual and nominal condition shows the extent of the requirements' fulfillment. This is essential for the customer satisfaction. Hence, the dimensions weighted level of performance and customer satisfaction are temporally behind the others and in the figure visualized shifted [12].

The multidimensional model has been implemented as a java-application. The models' dimensions which serve to structure the requirements and also the dimension "product" which serves to structure the reference object, to which the requirements refer, have been implemented in the system.

3 Functionalities of the developed requirements management system

During the planning process stakeholders interact with the data processing system not only because of a structured capturing of their requirements but also due to a requirements analysis (fig. 5). After the components and their software implementation have been specified the functionalities of the data processing system are described below.

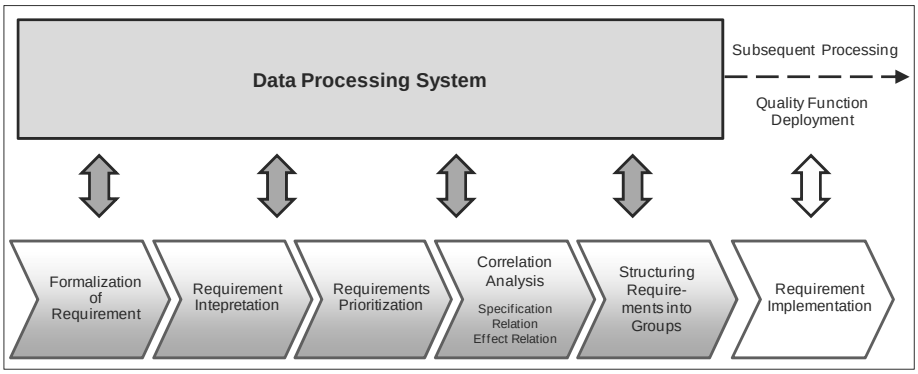


Figure 5: Functionalities of Data Processing System

3.1 Formalization and interpretation of requirement

After the stakeholder has formulated his requirement in natural language and entered it to the system, an automatic formalization of the requirement is performed by the linguistic analysis. Over a link to the wiki the stakeholder can check if the terms of his requirement were correctly interpreted by the system. In the case that the wiki contains more than one interpretation for a term, the stakeholder has to choose his meant interpretation. Requirements are checked regarding their meaning by matching the terms of the requirement with those of the ontology. Only meaningful recommendations for the formalization of requirements are provided. An incomplete match indicates that either a wrong requirement was set to the system or the knowledge of the ontology is deficient to understand the requirement. The stakeholder has the possibility to enlarge the ontology with knowledge by filing it with his terms and their interpretations as well as with missing properties between the existing classes. During these manual formalization, at first the stakeholder is displayed the reference-object-class-tree. As soon as a reference object is chosen or a new one has been added and an entity has been created, the stakeholder is asked, if it should be expressed that a reference object exists. That means an entity of the chosen reference object's class is generated and stored. Furthermore, the stakeholder has to decide, if he wants to place a requirement that states which reference objects consists of which other reference object(s). For this a tree of reference object classes is depicted for the user, with which the firstly chosen class of the reference object possesses a "consist-of"-relation.

After choosing or adding a reference object another entity is generated and stored in the system. Moreover, it is possible for the stakeholder to allot an attribute or a

value to the reference object. For this purpose, at first the attribute tree is opened which only shows attributes that can be chosen. Subsequently, classes of values which belong to the attribute's class are shown as a tree and a choice or addition can be made by the stakeholder. For storing a single requirement finally in the system, the other tabs of the template have to be filled with information, apart from the tab "constraints".

3.2 Requirements Prioritization

Within the large amount of given requirements contrary, competitive and excluding requirements will occur. Therefore, a weighting of requirements is necessary in order to determine an order of priority for requirements. The requirements prioritization can be defined by completing the tab "weighting" of the template (fig. 2). Thereby, the weighting of a requirement is the result of three sub-weightings.

The first sub-weight "importance of stakeholder" indicates which influence or competence the considered stakeholder has with regard to the product to be developed. If a requirement is e.g. prescribed by law, then the weight may be classified as high.

Within the second field "importance for project" it has to be specified which relevance the requirement has in the overall context of development project. For instance, the requirement on the realization of a decisive functionality can be more important than the requirement on the color of user interface in the context of product development.

The third sub-weight is the "weighting by stakeholder". Therefore, the stakeholder has to enter the weight of the considered requirement. Through this field it can be documented how important the implementation of a requirement is to the stakeholder. The priorities of the stakeholder can be captured out by simply setting of a weight by the stakeholder e.g. in the framework of an interview. Another possibility to collect priorities of an amount of requests is a conjoint analysis which often provides more effective results. The goal of a conjoint analysis is to establish a ranking of product bundles by stakeholder [13] with only a little effort. This enables the extermination of attribute part-worths [14] and thereby the weight determination of a requirement. One further possibility for a calculation of weighting is the method AHP (Analytic Hierarchy Process). This method uses paired comparisons to obtain weights to criteria and for the prioritization of alternatives [15]. In the framework of a subsequent correlation analysis (3.3) the system functionality "specification relation" can be used in order to transfer the amount of given requirements into a hierarchical structure (3.3) that is needed for the appliance of AHP. It has to be decided in the individual cases which method is the most appropriate for capturing the weights. It depends e.g. on the size of company or the experience of the staff. The appliance of conjoint analysis and AHP in the framework of the data processing system is part of further research activities.

3.3 Correlation analysis

Before a requirement can be entered entirely in the system by a stakeholder, an optional, automatic analysis by the system can be done after formalizing of the requirement. Thereby, relations between the regarded requirement and other single requirements can be depicted. Also, a differentiation between specification and effect relations can be made within the tab. Requirements are specified precisely during a dynamic development process [1]. Specification serves to clearly formulate the requirement demanded by making new requirements [3]. The subordinated requirements define precisely the superordinated requirements allocated to them [16]. Specification relations should provide automatically detailed requirements for the user, with which regarded requirements can be specified. This allows an optimization of expressing requirements by substituting abstract requirements with more concrete requirements. By this, a secured deriving of concrete requirements is enabled. Within the system a differentiation between object-, attribute and value-specification is carried out.

In the following, the specification of reference object will be described. This kind of specification relation is needed to concretize the reference object of a given requirement. The stakeholder can choose between more detailed reference objects that have an „is-a“-relationship or a „consists-of“-relationship to the object of his requirement before the output. An „is-a“-relationship is defined between the classes of the considered requirement´s reference object and their sub-classes. If he is choosing the „consists-of“-relationship the system outputs a list of all reference objects classes, to which the reference object of the considered requirement has a potential „consists-of“-property. In order to obtain a meaningful output a test was carried out to check whether the reference objects can have the considered attribute. During the specification of a reference object the system provides the stakeholder with more detailed reference objects.

The stakeholder can use the headline “constraint” of the template, if he wants to detect if and what kind of potential effect the setting of his considered requirement could have. Furthermore, he can find out with which other single requirements the considered requirement is connected. Effect relations and chains effect relations can be established by means of the defined “effects“-properties between the attributes. The stakeholder can see which other requirements compete with and which other requirements benefit from his requirement. A distinction is made between possible effects and effects to actually existing requirements. Based on this, the stakeholder has the possibility to revise or to cancel his requirement. Or he can strengthen his considered requirement by emission of further requirements that have a positive influence of the implementation of his requirement.

3.3 Structuring requirements into groups

The requirements which have been entered by using the template can be matched by the stakeholder to the dimensions and to the different categories of the dimensions. For this, the stakeholder can choose between the implemented dimensions for matching the regarded requirement to the chosen dimension. After that, for the respective dimension the different categories and sub-categories are shown and the stakeholder

can opt for the category he wants to sort the requirement in. This process can be repeated, if a requirement should be sorted in more than one dimension. Beyond that, matches of the same requirement which have been carried out by another stakeholder before are shown. This may assist the matching process and shows potential conflicts if requirements have been sorted in different categories of a dimension.

Furthermore, the system offers another assistance for matching "new" requirements to adequate dimensions and categories. Therefore, several search options allow looking for requirements that have been matched before by searching for example for similar terms to facilitate and optimise the structuring process. Also in this context, a statistical analysis is part of the system and offers assistance by showing terms to which requirements have been sorted in the system. These terms are linked to the respective requirements and the dimensions and categories they have been sorted in.

The developed requirements management system can show single dimensions of the model respectively single categories of dimensions and the corresponding requirements for the current situation. By this, it can be checked whether all stakeholders and their different requirements are surveyed.

3.4 Requirement implementation

The developed requirement management system serves as a preparation of requirements for later implementation within the scope of a QFD. Because it allows formalizing, interpreting, surveying and structuring of all the requirements on the reference object or rather the product. Structuring the requirements before using QFD is essential. Otherwise applying the method would not lead to valuable results, because of the multitude of the requirements which have to be considered and which (may) exist in different degrees of specification. For this it is possible to consider for example single groups of requirements which are single dimensions respectively categories of a dimension which show a similar or same degree of specification.

4 Advantages of the collaboration between the components of the developed requirements management system

Whereas the ontology and the template were developed to structure single requirements, the goal of the multi-dimensional model is to structure requirements in dimensions and categories into groups. Both parties benefit from each other. A check of the completeness and meaningfulness of a single requirement is executed by dint of the template and the ontology. The single requirements can be sorted into the model only after the requirements are formalized by means of the template. This supports that only complete and meaningful requirements are structured within the model for further use. In contrast to that, the model serves to check if all stakeholders and their requirements have been considered or if informational deficits exist. The model can be used to check if all requirements on the reference object have been surveyed. For this, single dimensions and categories with their respective requirements can be depicted.

Furthermore, it has to be considered, that the dimensions of the model are independent from each other. Knowledge about the relationships between the single requirements in the model is not defined. In contrast, the ontology contains rather detailed information about requirements and their correlation than the model. Due to that the ontology takes over the task of identifying relationships between requirements.

A high amount of terms located within the ontology exists that is needed to formulate single requirements. In contrast, the terms within the model are presented in the form of designations of dimension and categories. These designations serve as points of reference for the categorization of requirements. However, the designations within the model can be found as terms within the ontology. Due to that, the knowledge of ontology and the dimensions/categories of the model are compatible with each. In the future, the terms within the ontology will be summarized to tag clouds, which will be linked to the single dimensions and categories of the model. This could allow suggesting from the identified terms of requirements to their appropriate dimension or category of the model. The classification of requirements into the model could run (partially) automated. The stakeholder could have the possibility to confirm the proposal of the system and the result could be stored to the system. If inconsistencies appear in this described process, then this could be an indication of a wrong interpretation of requirement during the requirements formalization. This would serve as a further point of control that supports a correct emission of requirements. This function is a particular advantage when a requirement is set for the first time. If a stakeholder wants to sort a requirement the first time into the model, no statistical analysis is available. In this case, the stakeholder could rely on the defined link between the terms of his requirement and the designations of model.

The requirements' level of specification is considered within the ontology as well as in the model. However, there is a risk that requirements stay in the upper levels of the model and a detailed classification of requirements is not taking place. This complicates the check of requirements completeness within the single categories. The described function of specification relation could be used to solve this problem. A requirements specification within the template could give the information that it is necessary to check whether the detailed requirements have to be sorted into a more detailed level of the model. This would be the sublevel of the actual level of the original requirements category. It supports that requirements reach the appropriate detailing level of the model.

For every requirement the time of capturing is defined under the headline "situation" of the template. This information can be used in the time dimension of the model, in which the requirements' development over time is considered. Under the headline "situation" the status of requirement and its status of implementation is captured within the template. When a requirement has been implemented this will be noted as "implemented" within the template. That is the basis for the comparison of actual and nominal condition of a requirement by a stakeholder. The stakeholder can determine the extent of the requirements' fulfillment that is considered in the dimension weighted level of performance.

As the observations have demonstrated, it is essential to structure single requirements at first into a template and to structure them afterwards into groups. Consequently, both functions support each other's efforts. Furthermore, it was shown that a holistic requirements management is necessary which comprises in addition to gather-

ing and structuring of requirements the assessment of their implementation by the stakeholders.

Acknowledgements

The authors wish to thank the Deutsche Forschungsgemeinschaft (DFG) for supporting their work within the framework of the Collaborative Research Centre 696.

References

1. Jörg, M.-A., 2005: Ein Beitrag zur ganzheitlichen Erfassung und Integration von Produktanforderungen mit Hilfe linguistischer Methoden. Aachen: Shaker Verlag, 2005. ISBN: 978-38322-4032-5.
2. Herzwurm G., Schockert S., Mellis W.: Joint Requirements Engineering. QFD for Rapid Customer-Focused Software and Internet Development. Vieweg, 2003. ISBN 978-3-528-05736-7.
3. Humpert, A.: Methodische Anforderungsverarbeitung auf Basis eines objektorientierten Anforderungsmodells. Universität Paderborn : HNI Verlagsschriftenreihe, Band 9, 1995
4. Crostack, H.-A.; Kolbe, C.; Refflinghaus, R.: Relations between requirements on an intralogistics facility**.** In: Proceedings of the 12th International QMOD Conference (Quality Management and Organizational Development), 2009, Verona/Italien,27.-29.08.09
5. Eckstein, P.P.(2008) *Statistik für Wirtschaftswissenschaftler: Eine realdatenbasierte Einführung mit SPSS*, Gabler, Wiesbaden.
6. Borg, I.; Staufenbiel, T. (2007), Theorie und Methoden der Skalierung; Huber Verlag, Bern.
7. Borst, W. -N.: Construction of Engineering Ontologies for Knowledge Sharing and Reuse. http://purl.org/utwente/17864. [Zuletzt eingesehen: 02.10.2009.]
8. www.mediawiki.de. [Zuletzt eingesehen: 02.10.2009.]
9. Cimiano, P.: Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. Springer Verlag, Berlin, 2006, ISBN 978-038730632-2.
10. Gautum, N., Singh, N., 2008. Lean product development: Maximizing the customer perceived value through design change (redesign). Journal of Production Economics, 114, 313-332.3.
11. Crostack, H.-A.; Klute, S.; Refflinghaus, R., 2009: Structuring requirements on intralogistical-facilities - an approach. Proceedings of the 12th International QMOD Conference (Quality Management and Organizational Development), 27.-29.08., Verona/Italien.
12. Crostack, H.-A.; Klute, S.; Refflinghaus, R., 2010: A Multi-dimensional Model for Structuring Stakeholder Requirements, in: Proceedings of the 20th CIRP Design Conference, 2010, Nantes/France
13. Lilien G.-L., Rangaswamy A., De Bruyn A.: Principles of marketing engineering, DecisionPro, 2007
14. Wierenga B.: Handbook of marketing decision models, Springer, 2008
15. Saaty T.-L., Vargas L.-G.: Models, methods, concepts & applications of the analytic hierarchy process, Springer, 2000.
16. Krusche T. (2000): Strukturierung von Anforderungen für eine effiziente und effektive Produktentwicklung, Shaker Verlag, Aachen.

Focusing customer-orientation within requirements prioritization: the shape of Scrum as an agile software-development method

Benedikt Krams, Sixten Schockert

Department for Business Administration and Information Systems, esp. Business Software,
University of Stuttgart, Keplerstr. 17, Stuttgart, Germany
{krams, schockert}@wi.uni-stuttgart.de

Abstract. This paper provides several aspects why customer-orientation within the agile software-development method Scrum is not that distinct than it is proclaimed. After a short notice of the importance of customer-orientation, the demarcation of the term customer-orientation and a brief introduction to agile software-development methods the approach of Scrum gets discussed in the context of customer-orientation and requirements prioritization. This paper questions that approaches of agile methods as omnipresent customers during software-development projects or the customer-on-site practice are convertible for Scrum and applicable in practice. The consideration closes with an outlook and prospect for further research focusing on other agile software-development methods.

Keywords: customer-orientation, requirements prioritization, agile software-development, Scrum

1 Customer-orientation as a high value in software-development

By far customer-orientation is not a new concept. Some authors trace it back to the fifties and state it as a finding of marketing research [2]. On the other hand, some authors proclaim it as a rather new US-American management strategy [1]. Nevertheless, a successful concept does not lose its attraction and relevance only throughout existing for decades.

IT-companies and software-development firms point out their customer-orientation and probably no scientist or practitioner will be found negating the importance of customer-orientation, for products in general and as well as for software.[1]

The Chaos Report by the Standish Group, still often cited, states that user involvement is the number one success factor for software projects [24]. As will be shown in chapter 4, user involvement is very important for the agile software-development method Scrum, but its enforceability is very difficult in practice.

[1] E.g. the approach in [10]: Software Quality Function Models, pp. 29 and Prioritizing and Focused Software Quality Function Deployment, pp. 35; also: [15], p. 43

But is it true that during the requirements engineering process, especially for requirements prioritization, prioritization always happens customer-oriented?

Port, Boehm and Klappholz state that "there appears to be a lack of works that discuss requirements prioritization within agile development" [18]. This paper tries to close this gap partially for the agile software-development method Scrum. After the demarcation of the term customer-orientation and a brief introduction to agile software-development methods follows a discussion, which will provide several aspects why customer-orientation within Scrum is not that distinct as it is proclaimed.

The consideration closes with an outlook and prospect for further research focusing on remaining agile software-development methods.

2 The term customer-orientation

Literature emerged a lot of definitions of customer-orientation as far as the term has a long history. There are several authors who collected definitions of customer-orientation [22]. By studying these texts it can be seen, that there exists a wide range of shape. Recurring elements are:

- Collecting information from customers [21,13,19]
- Formulation of strategies [21,19] and long-term orientation [14,20,9]
- Satisfaction of needs [14,12,16,19]
- Organizational commitment [12,16,21]

Following we understand customer-orientation as a strategy. A strategy can be enunciated for a single person, an unit of an organization as well as for a whole organization [10].

Therefore we define customer-orientation as a strategy for the selection of action alternatives, which gives the target "satisfaction of the recipients of an outcome (satisfaction of customer needs)" the highest preference. It needs to be stressed that this definition implicates higher prioritization of e.g. quality[2] than cost reduction or efficiency targets.

3 Requirements engineering, requirements prioritization and software-development

As figure one shows, requirements engineering consists mainly of requirements analysis and requirements management. We focus on requirements prioritization which is one element of requirements analysis.

[2] Assuming, that a high quality leads into high customer satisfaction (e.g. [10], p. 15)

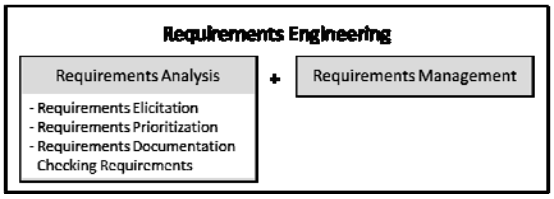


Fig. 1. Elements of Requirements Engineering

Requirements prioritization "helps to identify the most valuable requirements from (…) [author's notes: a set of requirements] by distinguishing the critical few from the trivial many" [5], assuming that most software-development projects have more candidate requirements than requirements that can be realized within time and cost constraints.

Following we take a look at agile software-development methods and examine the level of customer-orientation during requirements prioritization for Scrum, one example of an agile software-development method.

4 A (very) brief overview of agile software-development methods

Agile software-development methods have become more popular during time. Some of the most popular are Adaptive Software-Development (ASD), Chrystal Methodologies, Dynamic Systems Development Method (DSDM), eXtreme Programming (XP) and Scrum [23].

The goal of agile methods is to deliver products on time, on budget, with high quality and customer satisfaction. These goals are not distinct enough to differ between traditional software-development methods as far as they seem too universal. Sillitti and Succi say, that "the main difference between agile and traditional methods is the involvement of the customer in the development process" [23]; other authors stress the importance of the role of the customers for agile software-development methods as well [3,11].

Sillitti and Succi continue, that there is a set of practices derived from the "Agile Manifesto" [4] and stress amongst others the aspects 'requirements prioritization before every iteration' and an 'high customer involvement' [23].

This needs to be reflected and leads to the following question: Is Scrum a silver bullet when it comes to customer orientation?

5 Scrum and its level of customer-orientation during prioritization

First of all requirements get collected during requirements workshops [10,17] and it is clear that this is *the* point, where customers are involved into the development

process, regardless if software is developed with traditional or agile methods; therefore valid for Scrum as well. But how are customers involved *during* the process of the software-development with Scrum?

Pichler mentions that the product owner[3] is responsible for the prioritization of these requirements which are listed in the product backlog[4] and therefore needs distinct knowledge about customer requirements [17]; a hint for the absence of customers during the development process.

The Scrum team[5] does prioritization for its sprints[6] as well: team members decide which (in the product backlog prioritized) requirements get implemented first during one sprint: the team is autonomous in deciding how much work they reliably can do in that sprint [17].

Sillitti and Succi highlight the advantages of the so called customer-on-site practice: the recipient of the final product (the customer) stays with the development team *all the time*, is involved in the project and is *always* available to discuss with the development team. For example Broschinsky and Baker give an example of practical use of an agile method and state the presence of customers as well but without being more precise [7]. It seems to be allowed to doubt this omnipresence of the customer-on-site practice. Nevertheless Sillitti and Succi allude, that it is very difficult to implement a customer-on-site practice [23][7] and the drive of physically distributed software-development teams may not be able to accommodate the face-to-face communication advocated by agile software-development processes [26].

Ongoing the authors mention, due to practicability that one person needs to represent *all* stakeholders [23,8]. It needs to be questioned if one person is able to represent all stakeholders, especially external customers when not speaking of software-development for other e.g. business units.

6 Conclusion and outlook

So who is the real *prioritizator*?

When combing the understanding of the roles of the product owners and team members with the approach, due to practicability, of having one person representing all customers, it is not assured, that only the customers do the prioritization. In fact, especially during the software-development process, all kinds of people involved into the process prioritize requirements continuously.

---

[3] A product owner is responsible for requirement description and management (therefore for the product backlog; see the following footnote), releasemanagement, stakeholder management and at the end for project success ([17], pp. 9)

[4] A product backlog is the central instrument for collecting an managing requirements without containing activities for developers ([17], pp. 27)

[5] The Scrum team consists of engineers, software/business architects, tester, data-base specialists etc. ([17], p. 13)

[6] A sprint is a defined period of time for the scrum team to focus on a given list of goals and implement these goals.

[7] For premises of the customer-on-site practice see [23], p. 317

This conclusion can especially be drawn for Scrum as one agile software-development method based on the literature review done in this paper. For further research two points are necessary:

- Expand literature research for customer-oriented prioritization of requirements for Scrum as one agile software-development method.
- Take a closer look if the findings of this research paper can be adapted to other agile software-development methods.
- Ongoing analysis of the limitations of Scrum [25] for customer-oriented prioritization of requirements.

References

1. Albers, S., Eggert, K.: Kundennähe – Strategie oder Schlagwort, In: Marketing ZFP, 10. Jg., pp. 5-16 (1988)
2. Appiah-Adu, K., and Singh, S.: Customer-orientation and Performance: A Study of SMEs, In: Management Decision, vol. 36. no. 6, pp 385-394 (1998)
3. Beck, K.: Extreme Programming Explained: Embrace Change, Addison Wesley (1999)
4. Beck, K., Beedle, M., Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., Thomas D.: Manifesto for Agile software-development. Accessed on 5th of May 2010, online at http://www.agilemanifesto.org
5. Berander, P., Andrews, A.: Requirements Prioritization, In: Aurum, A., Wohlin, C. (Eds.): Engineering and Managing Software Requirements, Springer, Berlin (2005)
6. Boehm, B. W.: Software Engineering. In: IEEE Transactions on Computers. vol. 25, No. 12, 1976, pp. 1226-1241 (1976)
7. Broschinsky, D., Baker, L.: Using Persona with XP at LANDesk Software, an Avocent Company, Proceedings of Agile 2008 Conference, pp. 543-548 (2008)
8. Cohn, M.: User stories applied: for agile software development, Addison-Wesley (2004)
9. Deshpande, R., Farley, J., Webster, F.: Corporate Culture, Customer-orientation, and Innovativeness in Japanese Firms: A Quadrant Analysis, Journal of Marketing, vol. 57, pp. 23-37 (1993)
10. Herzwurm, G., Schockert, S., Mellis, W.: Joint requirements engineering: QFD for rapid customer-focused software and Internet-development, Vieweg, Wiesbaden (2000)
11. Hneif, M., Ow, S.H.: Review of agile methodologies in software development, In: International Journal of Research and Reviews in Applied Sciences, vol. 1, issue 1 (October 2009) (2009)
12. Hoffman, K. D., T.N. Ingram: Creating Customer-Oriented Employees: The Case in Home Health Care, Journal of Health Care Marketing, 11 (June), pp. 24-32 (1991)
13. Jaworski, B.J., Kohli, A.K.: Market Orientation: Antecedents And Consequences, Journal of Marketing, vol. 57 no. 1, pp. 53-70 (1993)
14. Michaels, R.E., Day, R.L.: Measuring Customer-orientation of Salespeople: A Replication with Industrial Buyers, Journal of Marketing Research, 22 (November) (1985)
15. Nassar, S.M., Barnett, R.: IBM Personal System Group: Applications and Results of Reliability and Quality Programs, In: Proceedings of Annual Reliability and Maintainability Symposium 2000, pp. 35-43 (2000)
16. O'Hara, B.S., Boles, J.S., Johnston, M.W.: The Influence of Personal Variables on Salesperson Selling Orientation, Journal of Personal Selling & Sales Management, 11 (Winter), pp. 61-67 (1991)

17. Pichler, R.: Scrum: Agiles Projektmanagement erfolgreich einsetzen, dpunkt, Heidelberg (2008)
18. Port, D., Boehm, B., Klappholz, D.: Nancy R. Mead: Making Requirements Prioritization a Priority, In: Proceedings of 21st Conference on Software Engineering Education and Training, pp. 250-261 (2008)
19. Ruekert, R.,: Developing a Market Orientation: An Organizational Strategy Perspective, In: International Journal of Marketing, vol. 9, pp. 225-245 (1992)
20. Saxe, R., Weitz, B.A.: The SOCO Scale: A Measure of the Customer-orientation of Salespeople, In: Journal of Marketing Research, 19 (August) (1982)
21. Shapiro, B.: What the Hell is 'Market Oriented'? Harvard Business Review, vol. 66, pp. 19-25 (1988)
22. Siguaw, J.A., Brown, G., Widing, R.E.: The Influence of the Market Orientation of the Firm on Sales Force Behaviour and Attitudes. In: Journal of Marketing Research, vol. XXXI (February 1994), pp. 106-116 (1994)
23. Sillitti, A., Succi, G.: Requirements Engineering for Agile Methods, In: Aurum, A., Wohlin, C. (Eds.): Engineering and Managing Software Requirements, Springer, Berlin (2005)
24. Standish Group: Chaos Report Reprint 1995. Accessed on 5th of May 2010, online at http://www.projectsmart.co.uk/docs/chaos-report.pdf
25. Turk, D., France, R., Rumpe, B.: Assumptions Underlying Agile Software Development Processes, In: Journal of Database Management, vol.16, no. 4, pp. 62-87 (2005)
26. Turk, D., France, R., Rumpe, B.: Limitations of Agile Software Processes, In: Proceedings of the Third International Conference on Extreme Programming and Flexible Processes in Software Engineering (2002)

Requirements Engineering and Service Commitments

Alexander Rachmann, Irene Maucher

Niederrhein University of Applied Sciences,
Webschulstraße 41-43, 41065 Mönchengladbach, Germany
Alexander.Rachmann@hs-niederrhein.de

T-Systems International GmbH,
Am Propsthof 51, 53121 Bonn, Germany
Irene.Maucher@t-systems.com

Abstract. We are introducing service commitments as one artifact in the requirements engineering beside goals, scenarios and solution-oriented requirements. We explain why to model commitments, how the structure of the commitments and the goals should be related, and how to model commitments in relation to other artifacts.
We describe a case study with two scenarios and relate these scenarios to both goals and service commitments. A conclusion closes this paper.

Keywords: requirements engineering, service engineering, service, service commitment, requirement artifacts

1 Requirements Engineering and Service Engineering

In one of our current projects where we are developing AAL-services (see case study), we noticed that the customers found it more useful to prioritize service commitments than goals of the service. Due to this phenomenon, we decided to focus on the commitments. However, the state-of-the-art literature in service engineering does not include service commitments as an artifact. An integration of service commitments in Requirements Engineering is needed.

Service Engineering (SE) is the systematic development of services. This fairly young engineering discipline has a strong relationship to software and systems engineering. The phases analysis, requirements definition, conception, test and implementation are commonly accepted [1]. There are clear analogies in these phases to software engineering: the rough structure from requirements to conception, followed by testing and implementation.

However, not all methods and techniques can be translated from software engineering to service engineering as simple as that. We will see how the requirements engineering, as part of the service engineering process, must be adjusted to service engineering.

To understand the basic differences between software and service engineering, one has to understand the difference between software and a service. End-Consumer recognize Software most often as a product -- one buys a piece of software (online or in a real-world store) and owns the software from thereon. Of course, there can be service-add-ons, such as regular updates, etc., which bring the software to a more service-like thing. Also, this does not account for software engineered and / or deployed for an individual organization. There is a trend towards deploying software as a service (SaaS). However, today, standardized software is mainly seen as a product. In the German marketing discipline was in the mid 90s an ongoing discussion about the differences between a service and a product. As a result of this discussion one agreed on the following definition of a service: A service is an independent, market-ready effort. For this effort, one allocates and / or provides resources, thereby combining the resources of the provider and the customer. The effort is meant to be useful to humans [2].

A service commitment is a promise, made by a service provider to a customer at the time of the acquisition. As in the definition of service, this commitment is meant to be useful to the customer. It manifests in a (written or verbal) contract.

Prioritizing is the act of giving relevance to an artifact in comparison to other artifacts of a certain domain [3]. This relevance is based on criteria to be addressed within the artifacts, such as importance, costs, damage, volatility or risk. It is advised to start prioritizing the most abstract artifacts (in classic RE: goals) and then go further to the more concrete artifacts (scenarios and solution-oriented requirements). However, service commitments are not yet integrated in RE. This paper is a first start to do so, in order to base the prioritization of service commitments on a theory-grounded foundation.

2 Service Commitments

We are regarding to the requirements engineering framework of Klaus Pohl [3]. He differentiates three main artifacts: goals, scenarios and solution-oriented requirements. A goal is the intentional description of a characteristic of the engineered system resp. the according engineering process. In SE, the engineered system is a service, i.e. a socio-technical system. A scenario describes a concrete example concerning fulfillment or non-fulfillment of a goal. A solution-oriented requirement is a condition or property of a system to solve a problem or to achieve a goal (see also IEEE Standard 610.12-1990). These solution-oriented requirements are most often seen from the function, data and behavior perspective. The relations between these three artifacts are depicted in figure 1: A scenario substantiates one or more goals. A requirement substantiates one or more scenarios. This can be depicted as a three-step ladder.

However, we want to establish a fourth artifact that is the service commitment. The service commitment is placed parallel to the goal. A goal delivers the service commitment; a scenario verifies the service commitment, more precise: verifies how a service commitment is delivered. A service commitment is a promise made by a

service provider to a customer at the time of the acquisition. It manifests in a (written or verbal) contract.

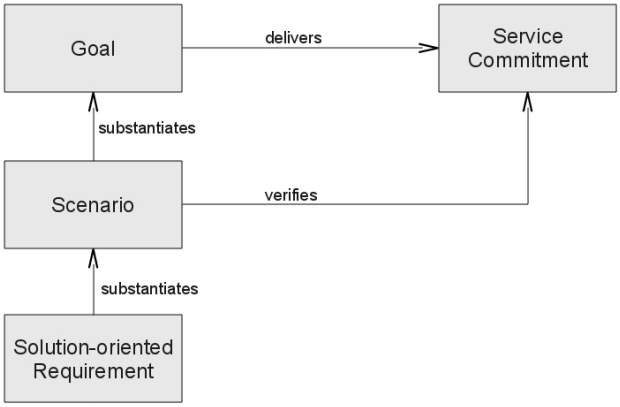


Fig. 1. Metamodel of goals, service commitments, scenarios, and requirements

Goals relate to each other. They can be part of one another and / or depend on one another (support, constrain, foreclose, or be equivalent to one other). With the service commitment, the provider makes the promise to fulfill one or more certain goals. However, there are three main issues:

1. why to model commitments, goals or both artifacts,
2. how the structure of the commitments and the goals should be related, and
3. how the process of modeling the commitments and goals works.

2.1 Commitments and goals

First, as already defined, a goal is the intentional description of a characteristic of the engineered system / the according engineering process. This implies a high probability that the system (be it a product whatsoever) can and will achieve this certain goal. Most often, this probability is best controlled, if all required actions lie with the manufacturer of the system. However, one distinctive characteristic of service is that the customer is involved in the service delivery. Without the activity of the customer the service can -- most of the time -- not be delivered. Also, most services are constrained by laws which might also inhibit the achievement of the goal: This might be the case, if the need of the customer is contrary to laws.

Second, a commitment formulates the process of service delivery, whereas a goal defines the results, i.e. the value for the customer. Having a scenario with no obvious value at first sight, it seems intuitive to describe what one can do (service commitment), before stating what value is being delivered (goal). However, at the end of the requirements process, the engineers and service providers must be able to state both, service commitment and goal.

Third, a goal implies a certain state of the system at a certain point in time; most often this certain state is an advancement for the customer. As in health care one does not always focus on the advancement of state, but rather tries to avoid the decline of a certain state, e.g. in palliative care or in some nursing care. It seems more intuitive to formulate this as a commitment rather than as a goal: The commitment would be, that the provider does everything he can do to hold a certain state. But without the help of the patient, his actions will not be successful. In this case, he will fulfill his commitment, but does not achieve the related goal.

Table 1 gives an overview of these three arguments concerning commitments and goals.

Characteristic	Commitment	Goal
Control	Provider and Customer	Provider
Reference	Process	Result/ Value
Change of State	No	Yes

Table 1. Characteristics of commitments and goals

2.2 Structural Analogousness between Goals and Commitments

It seems intuitive to model the goals of a system in a hierarchy with only one goal on the top, making all other goals sub goals of the nth level of the top goal.

However, it is of crucial importance how to model the service commitments. Obviously, there are parallels between the goals and the commitment. The service commitments can also be seen as a hierarchy, they can also be related to the scenarios. It seems advisable to relate the main goal (i.e. the top goal in the hierarchy) to the main service commitment. It must be further examined if it is advisable to model a structural analogousness between goals and commitments, as depicted in figure 2.

In this figure, G1 is the top goal of G1.1 and G1.2; furthermore G1.1.1 and G1.1.2 are an or-decomposition of G1.1. That means, G1 can only be fulfilled, if G1.1.1 or G1.2 are fulfilled. G1.1.2 constrains the fulfillment of G1.2. The service commitments are modeled analogue: SC1 is departed in SC1.1 and SC1.2, etc. One could relate these three levels to the well-known levels of a service level agreement (SLA). However, further research is needed therefore.

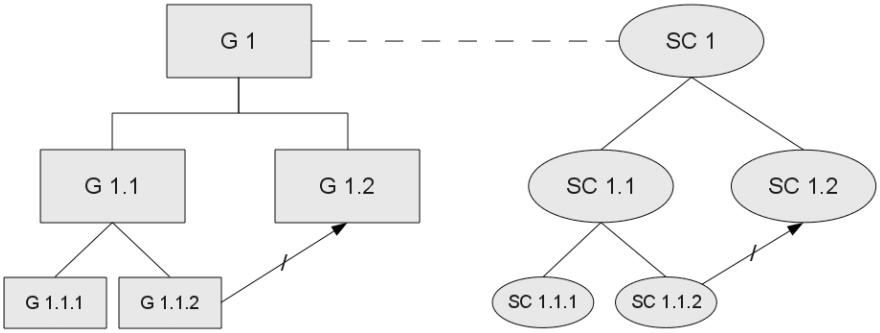


Fig. 2. Analogue Structure of the Hierarchies for Goals and Commitments

To us it is still unclear if the structures should be analogue. If so, it is not clear if all layers of the hierarchies are modeled in both structures. A compromise could be, to model for the top and second level goals according service commitments. These commitments could be easily communicated via marketing and could be manifested in a contract.

2.3 Process of modeling the service commitment

As it is commonly agreed, Requirements Engineering is a cooperative, iterative and incremental process. In the case of goals, scenarios and commitments, one has to find the best suited starting point.

Goals are best suited, if the provider is mostly independent in achieving this goal. On the other hand, service commitments are useful, if the provider can assure them parts of the service delivery, but not if the service will have the intended outcome. These restrictions in delivering the service may be due to missing involvement of the customer, legal constraints or superior force. Service commitments lay the focus on how to deliver the service, rather than on the results i.e. the value for the customer.

It is well known that scenarios are a good starting point if the service provider does not know exactly what the needs of the clients are. By describing scenarios, i.e. a concrete description of a situation of a customer in need, the service provider is able to understand what he can do for the customer. In this context, the service commitment is an abstract version of the involvement of the service provider: not only what the service provider can do for the client, but rather what they can assure.

For us it seems advisable to start with scenarios, then take on service commitments then goals. All steps should be iterative and incremental.

3 Case Study

We are regarding two scenarios in this case study. Both are about the usage of Ambient Assisted Living technologies (AAL), i.e. the use of modern micro sensor in the direct environment of people to support their daily lives. The first scenario describes an existing service which is migrated to a modern AAL-service. The second scenario describes a fall detection system (which is right now not yet available on the market).

3.1 "I am doing well" message

In the care for older and single people it is common to use a signal between the person and the nurse. However, this is usually quite non-technical: one may use a turnable sign on the door, the one side red, the other one green. If the person needs (non time critical) help during the day, he may turn this sign, and the nurse will understand to visit that person. The disadvantage of this classic implementation: the nurse and the person must be somewhere near that sign.

The modern AAL-technology is quite easy implemented: One installs a button in the residence of the person. As soon as he needs help, he pushes the button and the nurse will come.

The related service commitment would be, to come as soon as possible. The related goal would be that the nurse is as soon as possible with the person. However, this goal can only be achieved with the critical involvement of the customer. For the service provider the formulation of he service commitment is safer: that describes what they are able do.

3.2 Fall detection

As already mentioned, there is no system on the market to detect the fall of a person. However, research and technology are progressing towards the goal of being able to detect falls. Assuming that the residence is equipped with the appropriate sensors, the scenario would be as follows: The system receives information and interprets it as a fall; the system then requests a response from the resident. There can be three cases:

- The resident has fallen and is not able to move, or
- the fall was not critical and the person is up again, or
- the resident did not fall at all.

If the resident is not up again and does not respond to the systems request, an alarm is activated to the service provider (i.e. nursing station).

In this case the service commitment would be that the service provider calls an ambulance as soon as possible. The goal would be that the person gets help as soon as possible. Again, the service provider can not achieve this goal by himself; they are dependent on the involvement of the resident (i.e. pushing or not pushing the alarm-button) and the ambulance (i.e. how fast the ambulance is at the scene).

4 Conclusion and further research

We describe the basis of an alternative approach to requirements artifacts. Beside the well established goals, scenarios and solution-oriented requirements, we placed the service commitment. A service commitment is a promise, made by a service provider to a customer at the time of the acquisition. It manifests in a (written or verbal) contract. It seems advisable to use the same structure for service commitments in a (service engineering) model as well as in a (law relevant) contract.

For certain domains, it seems intuitive to work with commitments rather than with goals. However, the contrary is also true: there are certain domains where it is advisable to stick to the goal-oriented approach. We will further investigate

1. under which circumstances service commitments are better suited than goals,
2. how the structure of the service commitment is related to the structure of the goals, and
3. how the modeling process is best designed to fit service commitments, goals, and scenarios, especially in prioritizing the artifacts.

Further results of this study are to be published in a project report by T-Systems in 2010.

5 References

1. Bullinger, H.-J., Schreiner, P.: Service Engineering: Ein Rahmenkonzept für die systematische Entwicklung von Dienstleistungen. In: Bullinger, H.-J., Scheer, A.-W. (eds.) Service Engineering -- Entwicklung und Gestaltung innovativer Dienstleistungen, 2. edition, pp 53-84. Springer, Berlin, Heidelberg (2006)
2. Meffert, H., Bruhn, M.: Dienstleistungsmarketing. 6. edition. Gabler, Wiesbaden (2009)
3. Pohl, K.: Requirements Engineering. 2., revised edition. dpunkt, Heidelberg (2008)

Integrating Prioritization into Business Process-driven Requirements Engineering

Norman Riegel[1], Sebastian Adam[1] and Oezguer Uenalan[1]

[1]Information and Interactive Systems Department
Fraunhofer IESE
Fraunhofer-Platz 1
67663 Kaiserslautern, Germany
{norman.riegel, sebastian.adam, oezguer.uenalan}@iese.fraunhofer.de

Abstract. One of the major decisions to be made in any IT project is the prioritization of requirements to be implemented. The prioritization techniques in the literature, however generally do not consider the differences in the nature of requirements on different levels of abstraction, making it hard to guide requirements refinement in IT projects based on priorities. Furthermore, most techniques are based on the subjective assessment of different stakeholders and do not calculate the actual business value of a requirement. In this paper, we therefore propose a prioritization method for IT projects that refines and prioritizes requirements in a systematic and more objective way, from an enterprise's business processes down to the system function level. Therefore, we combine business process-driven requirements engineering with prioritization issues.

Keywords: Requirements prioritization, requirements prioritization method, business process, information system, business process-driven requirements engineering

1 Introduction

IT projects in enterprises are typically aligned with business process improvement programs aimed at optimizing business performance. There is a continuous cycle of measuring business performance and selecting processes to be improved, including corresponding adaptations in the IT landscape. When planning improvement programs, managers typically need decision support in order to assess the best way to spend the available time and budget on an improvement action. In the area of requirements engineering and release planning, prioritization is an established strategy used for such goals. Basic prioritization criteria include cost as a restriction, or as a factor to be minimized, on the one hand, and the benefits entailed by the implemented requirements as a factor to be maximized on the other hand. In the literature many prioritization techniques have been proposed, differing in terms of their complexity, the scales used, the stakeholders involved or in their input and output [5][6]. The techniques have in common that they take as input flat requirements, requirements groups or requirements hierarchies, but only give little

guidance on how to derive and refine the requirements as it is usually done in requirements processes. Even if this not the per se purpose of prioritization, the consequence is that requirements might get different priority depending on which group they belong to or which hierarchical level they are on. At the same time, most prioritization techniques solely depend on the subjective assessment of different stakeholders. The stakeholders have to state their opinion regarding the importance of certain requirements or features without knowing what importance really means, i.e., they do not know which value they should assign to a requirement. This is especially the case in non-comparative techniques where priorities are based on simple scales ranging, for instance, from “not important” to “very important”.

In this paper, we propose a prioritization method for IT projects that refines and prioritizes requirements in a systematic and more objective way, starting from an enterprise’s business processes and going down all the way to the system function level. Therefore, we combine business process-driven requirements engineering with prioritization issues, as refinement of requirements and prioritization should be done in an integrated manner. The paper is structured as follows: Chapter 2 gives an overview of related work. Chapter 3 presents our approach on how to derive the different requirements artifacts as well as the prioritization guidance. Chapter 4 gives a conclusion and outlook on future work.

2 Related Work

A commonsense opinion regarding the prioritization of requirements is that only requirements on the same level of abstraction should be prioritized together [2][9]. Approaches like cumulative voting (CV), the Kano model [3], Wieger’s method, [9] or simple Likert scales, which do not take into account hierarchical levels, therefore do not provide guidance on how to derive requirements on a reasonable level of abstraction. Linking and prioritizing requirements on different levels is not new, it was already used in the QFD approach by Akao [10]. Hierarchical Cumulative Voting (HCV) [2] as a hierarchical approach divides requirements into high-level and low-level requirements, making it possible to create requirements trees. However, HCV gives no guidance on how to break down the requirements into different levels. The cost-value approach [7] is also capable of taking into account requirements hierarchy levels by making use of AHP [8]. But again no guidance is given on how to build up reasonable hierarchies. Furthermore, these approaches do not consider the nature of the different levels (e.g., requirements on business process level vs. on system function level), which might result in unsuitable prioritization criteria. In [5] and [6] a generic framework for classifying prioritization methods based on benefit and cost estimation is proposed. This framework shows that prioritization is generally done on an already specified set of requirements, i.e., that derivation of requirements artifacts and prioritization techniques are usually not intertwined. The consequence is that requirements refinement cannot benefit from an earlier prioritization, which also makes it hard to define increments and optimize the order of elicitation during the requirements phase already. With respect to the support given to the stakeholders regarding how to assign specific values to the requirements artifacts, [9], for example,

provides some hints on which factors could be relevant for giving a certain value. But the factors mentioned are too fuzzy to derive a concrete value on a scale. Thus different stakeholders would choose different values, which in turn would not be objective.

3 The prioritization approach

The derivation of requirements in IT projects is often based on the business processes to be supported. To provide more guidance in this derivation process, we have developed a conceptual model in our previous work [1] that clarifies how requirements on different levels of abstraction are related in such a setting. To illustrate this model exemplarily, Fig. 1 shows how it is typically applied.

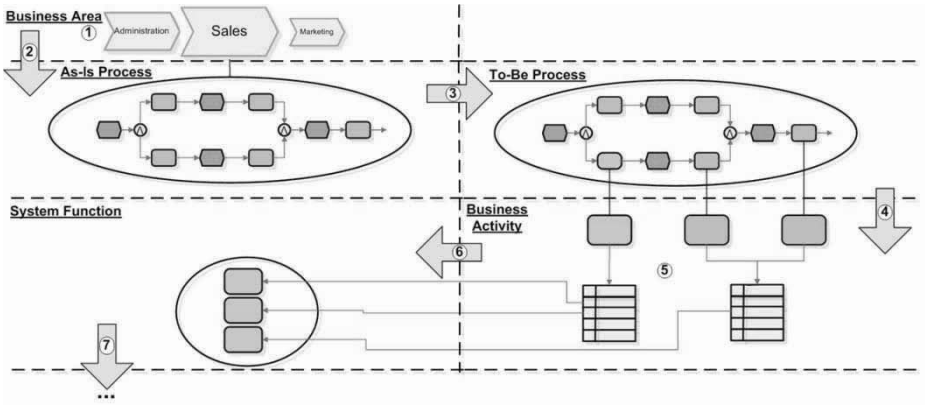


Fig. 1. Steps of our envisioned prioritization method.

A business area (step 1) "Sales" could be selected in a company in order to be improved. An as-is process (step 2) might be the process "receiving orders". Based on this, an improved to-be version for this process would be derived (step 3), e.g., by providing new channels to receive orders. From this to-be process, a set of business activities is derived (step 4), e.g., "create an order". Each activity consists of steps (step 5) in order to execute this activity. Such steps could include "Login" or "Fill out Order Form", for instance. All steps to be performed with an IT system are then specified by system functions (steps 6 and 7). The prioritization method presented below is based on this conceptual model. Basically, the step of identifying conceptual element instances on a certain level of abstraction, the prioritization of these instances and the subsequent refinement of a next level have to alternate when following our approach.

Step 1: Identify relevant business processes

Each improvement program aims at a certain business area to be improved. A business area is logical segment of an organization depending on the markets, etc. it

acts in. The identification of all business processes within this logical segment is therefore the first step.

Step 2: Prioritize as-is business processes

After having identified all main business processes of the business area of interest, the processes are prioritized. In this step, the processes are still regarded as black boxes, i.e., the flow of business activities within the processes is not yet regarded. The purpose of this prioritization is to filter the business processes that should be further considered. For prioritizing the business processes, the following two-dimensional scheme is used: The first dimension is the business value (BV), which quantifies how important a process is for the business segment. We propose calculating this value using objective measures as follows:

$$BV_{process} = \frac{Type \cdot \log_{10}(MIN(Executions\ per\ month + 1, 1000))}{3}$$

The type of process is expressed through numeric values with a precisely defined meaning. If the process is indispensible to realizing the business' services, i.e. within the value-chain of the company, then the value of *Type* is 3. If the process is a support process necessary for the internal operation of the enterprise (e.g., administration), then the value of *Type* is 2. Optional processes that are just used for achieving a self-defined quality level have a *Type* value of 1. Besides the type, also the number of executions of the process needs to be considered. However, to make the processes more comparable, the *log* is used, as it only increases slowly for larger numbers of executions. Furthermore, we use MIN to normalize the BV between 0 and 3[1]. If completely new processes are created, then the values for the formula above should be estimated.

The second dimension describes the need for change (NC), which quantifies how important it is to change the way the process is currently being executed. The NC is expressed through numeric values; again with a precisely defined meaning. If the process must be changed by law, the NC is quantified with 3 (must change). If there are weaknesses in the process that should be improved due to strategic decisions to remain competitive, the NC should be quantified with 2. If there is a possibility to change a process without an urgent need, the NC should be 1. In the event that a process should not be changed, the value should be 0.

The result is displayed in a two-dimensional diagram (see Fig. 2). The x-axis reflects the need for change, while the y-axis reflects the business value of the process. To determine the sequence in which the prioritized business process should be refined, the following rules are applied: all processes belonging to sector A must be further refined and considered, as they must change. Processes belonging to sector B should be refined as they have a medium or high business value and some improvement need. Processes belonging to sector C should only be refined after all processes of sector B have been refined (and there is still time and budget). For processes belonging to sector D, it is questionable to further refine them, as they only have a medium business value and do not have an urgent need for change. Processes in sector E should not be further regarded.

[1] At the moment, the function is normalized to 10^3, which might not be optimal for all projects or domains (e.g., in a travel agency, there might be a lot more than 1000 process instances per month).

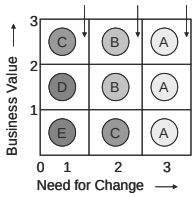


Fig. 2. Prioritization matrix.

Step 3: Specify to-be business processes

In this step, the selected business processes are specified in a desired to-be state. To make this happen, the as-is state of the selected processes is taken as input. If no as-is processes exist yet, they can also be modeled in advance. During the definition of the to-be process, it is also determined which business activities will be supported by the intended information system and which will not.

Step 4: Prioritize business activities

Each business process consists of a sequence of business activities that need to be performed in order to execute the business process. All activities that will be supported by the intended information system are now regarded in this step (still as a black box) and prioritized according to the scheme already used step 2. However, there are differences, regarding the calculation of the BV. When calculating the BV of a business activity, the overall number of executions per month must be calculated. If, for instance, a business activity is used in more than one selected business process, the sum of the number of executions of each process must be taken to calculate the overall number. However, the probability that a business activity is actually executed within a process instance must also be considered. For example, if a business process has alternative paths, the probability of running through each path has to be specified. The types of business activities are similar to the process types: If the activity is indispensable to create the process' outcome, i.e. it is indispensible to produce the business value of the process, the value of *Type* is 3. If the activity is a necessary support activity but does not contribute to the business-value creation of the process (e.g., an administrative activity), the value of *Type* is 2. Optional activities that are just used for achieving a self-defined quality level have a *Type* value of 1 (e.g., feedback sessions). Hence, the business value BV of an activity can be calculated as follows:

$$BV_{Activity} = \frac{Type \cdot \log_{10}(MIN(\sum_{p=1}^{n} probability_{Activity_p} \cdot Execution\ per\ month_p + 1{,}1000))}{3}$$

Through the creation of the to-be processes, new activities can emerge or old activities might be merged. Hence, the NC for these activities will be set to 3. For all other activities the same rules for the NC are applied as in step 2. Also, the selection will be done according to the matrix shown in Fig. 2. The prioritization of the business activities does not take into consideration the priority of the business processes they are part of because there is no one-to-one relationship between both. As already mentioned, the prioritization of business processes is rather used for filtering and focusing the requirements refinement and not for immediately selecting the requirements to be implemented.

Further steps:
In principle, the approaches described above can be applied for the lower levels too, i.e. specifying business activities (typically as use cases [4]), prioritizing system functions etc., but here we have to give full particulars in future work. At this level, a reasonable cost-estimation could also be performed and juxtaposed to the value derived by the previous steps. Here, release planning can be done on a detailed level.

4 Conclusion and future work

In this paper, we have proposed first ideas regarding a prioritization method for IT projects that shows how to refine requirements artifacts, from business processes down to the system function level, in a systematic way by prioritizing them in a more objective manner by using measurable criteria. Some open issues remain, which will guide our future work. First of all, the prioritization functions and dimensions are subject to optimization. There might be better functions for calculating the prioritization value than the *log*, for example. The business value might also be calculated by integrating monetary values. Also, the need for change dimensions might be extended in order to be more objective and more fine-grained. Also, a cost dimension should be regarded. Finally, we want to extend the method to include more aspects of our information system conceptual model [1] and, in particular service-oriented artifacts.

References

1. Adam, S., Naab, M., Trapp, M.: A Service-oriented View on Business Processes and Supporting Applications. To appear in: Proceedings of 11th Workshop on Business Process Modeling, Development and Support, Hammamet (2010)
2. Berander, P., Jönsson, P.: Hierarchical Cumulative Voting (HCV) - Prioritization of Requirements in Hierarchies. International Journal of Software Engineering and Knowledge Engineering 16, vol. 6, pp. 819--849 (2006)
3. Berger, C. et al.: Kano's Methods for Understanding Customer defined Quality. Center for Quality of Management Journal 4, vol. 2, pp. 3--36 (1993)
4. Cockburn, A..: Writing Effective Use Cases. Addison-Wesley, Boston (2000)
5. Daneva, M., Herrmann, A.: Requirements Prioritization Based on Benefit and Cost Prediction: A Method Classification Framework. In: 34th Euromicro Conference Software Engineering and Advanced Applications, pp. 240--247, (2009)
6. Daneva, M., Herrmann, A.: Requirements Prioritization Based on Benefit and Cost Prediction: An Agenda for Future Research. In: 16th IEEE International Requirements Engineering Conference, pp. 125--134, (2009)
7. Karlsson, J., Ryan, K.: A Cost-Value Approach for Prioritizing Requirements. IEEE Software 14, vol. 5, pp. 67--74 (1997)
8. Saaty, T. L.: The Analytic Hierarchy Process. McGraw-Hill, New York (1980)
9. Wiegers, K. E.: First Things First: Prioritizing Requirements. Software Development 7, vol. 9, pp. 48--53 (1999)
10. Akao, Y.: Quality Function Deployment: Integrating Customer Requirements into Product Design. Productivity Press (1988)

Improving IT-Strategy-Alignment and requirements engineering with a multi-dimensional business value

- Research Preview -

Andreas Rusnjak

Christian-Albrechts-Universität zu Kiel, AG Angewandte Informatik (Wirtschaftsinformatik),
Hermann-Rodewald-Str. 3, 24118 Kiel, aru@informatik.uni-kiel.de

Abstract. The number of e/mCommerce user as well as the number of companies and new business models is growing permanently in internet-based markets. These markets are affected by a high dynamic and a high sales-potential. Therefore project management and software development is becoming more and more important for technology-based companies in order to deal with the dynamics of fast-paced markets and to keep or strengthen their market position. Many e/mCommerce-Projects are failing due to insufficient planning and information, barely existing strategic alignment, poor management, conflicting ideals and objectives between all involved stakeholders and the noninvolvement of (key) stakeholders. In order to deal with these conflicts, we need to manage these projects with a better strategic alignment as well as using easily understandable indicators over all hierarchical levels of enterprises. In our model, a first approach, domain values provide support for goal- and value-based software development which seems to enable better prioritization over multiple business domains, enhanced focus on strategic goals and better understanding of market needs.

Keywords: Business Value, Requirements, Prioritization, Strategy, Alignment

1 Initial situation, concurrent perspectives & failure of projects

1,802,330,457! That's the number of Internet users worldwide on 12.04.2010 [1]. Not only the number of people constantly entering new dimensions, but also the number of companies which are using the internet to implement new business models. In 2008 42 percent of the German people bought goods online. In the European comparison Germany is taking the fourth place behind Great Britain (49%), Denmark (47%) and Netherland (43%) [2]. Compared to 2008 the annual online-turnover of goods in Germany will increase up to expected 15.4 billion Euros (+ 15%) in 2009. Besides the shippers the service providers are also taking profit from an increasing trend. The turnover of music downloads, online ticketing and bookings will increase from 5.9 billion Euros (2008) up to expected 6.4 billion Euros in 2009. With 21.4 billion Euros total online-turnover the eCommerce in Germany seems to defy the economical crisis. [4]

One more perspective with an increasing significance for companies is the world of mobile technologies. In 2008 the number of cellular mobile contracts in Germany (circa 100 million) is higher than the number of residents with circa 82 million [5, 6]. The rapid distribution of cell phones and more mobile devices (e.g. PDAs, Smart Phones, Netbooks, etc.) is offering companies more possibilities for additional business models or success stories in mobile business (c.f. [7]). Due to the popularity of eCommerce and chances of mCommerce more and more enterprises are gaining their turnover in electronic markets. They are entering the combat for market shares and need to cope with high levels of complexity in managing fast paced social, regulatory and technical developments to increase the number of customers and the turnover. In comparison to the competitors companies needs innovative procedures as well as a fundamental knowledge about the requirements of their organization and environment with a focus on Critical Success Factors (CSFs). CSFs are factors with a significant influence to company success [8]. As a result the majority of innovative business models are technology-driven and customers in digital markets access companies predominantly via software-interfaces, e.g. a website. Because of that and due to changing consumer behavior, a technology- and innovation-orientation as well as an efficient Project-Management (PM) and software development are becoming more and more important but it's not a perfect process by itself. McLaughlin (2009) is showing in his case study typical problems causing the failure of software development projects. The problems were (i) ambiguous objectives, (ii) unrealistic goals, (iii) unclear references to strategy, (iv) poor communication and (v) an insufficient leadership. In addition, concerned stakeholders were not involved in the formulation of requirements and not involved during the realization. The project was mostly driven by technical employees without any exact knowledge of the real requirements of the stakeholders/ market. [9]

More studies also reveals shortcomings like (i) ambiguous tasks and deadlines, (ii) poor information (missing, wrong or out-dated), (iii) ambiguous responsibilities, (iv) an insufficient document management and (v) poor communication [10, 11]. The reasons for the failure of eCommerce-Projects are various. Both empirical experiments as well as scientific work are showing that most of the reasons are insufficient planning (time, costs, and resources), poor management and different ideals and goal objectives of the involved stakeholders. In order to successfully manage eCommerce-Projects all stakeholders need to understand the vision of the project, the strategic goals and the ideals and goal objectives of all concerned parties. Management-Support is a key factor for a successful realization of eCommerce-Projects or implementation of eCommerce-Systems. It helps to emphasize the need for technology or innovation and obtain strong commitment from all involved parties in the project. If top management doesn't provide a clear direction or vision, involved stakeholders may get confused and projects will fail. [12, 13]

Due to this we're working on a multidimensional business value model - from a company's internal perspective - to improve the process of customer focused value performance from prioritization aspects in (early) requirements engineering up to delivering final software solutions for technology-driven companies in eMarkets.

2 About business value

The Value contributed by IT is playing a significant role in eCommerce/mCommerce-Business. Most of IT-Projects are traditionally focused on delivering functionalities, features or services to business. Focusing on business value will provide some key benefits like (i) a better alignment between strategy and IT, (ii) a change of IT from service provider to partner and (iii) a description how IT is satisfying the business' fundamental goals and rapid changing requirements [14]. Mahmood et al. (2008) state that there's "*little or no empirical research in ecommerce business value, but some related concepts already identified include business value; e-commerce impact; and e-commerce businesses success and failure.*" [15].

We agree on this and will roughly describe the business value concept as a base for later discussion. Defining business value seems to be a difficult task. In order to do it adequately, it is imperative that one appreciates the variety and complexity of factors that determine business value and those that influence it within an organization. Williams & Williams (2003) define business value (of an investment) in economic terms as "the net present value of the after-tax cash flows associated with the investment" [16]. Matts & Pols (2004) have identified a possible creation of business value from a certain project when "*it increases or protects profit, cash flow or return on investment in alignment with the company's strategy*" [14]. Tosic et al. (2007) recognize the business value as "*a broad concept that refers to any measures of worth of business entity. It includes not only financial aspects (e.g., income, costs, profit) but also many other aspects (e.g., market share, customer satisfaction) important for business operations*" [17]. Business value = f(cost, time, functionality, quality) [18]. The meaning of business value, depending on one's perspective, spreads out into different dimensions of both tangible and intangible values with structural significance to the different stakeholders [19]. Its implementation requires both, financial assets and human resources that can guarantee its achievement and steer it in the right direction. Considering the fact that the business value of an organization depends on numerous influences, e.g. the level of information or environmental issues that are dynamic in their nature, it would be easier for management to deal with a model that has assumptions, input and output, instead of using some prognosticated statements. Possible determinants for success of eCommerce and part of business value are performance, productivity and perception (e.g. companies image and customer satisfaction).

Performance is measured by financial indicators (hard factors) like return on investment, return on equity, return on sales, growth in revenue, etc. and productivity in sales to total assets, total sales and sales by employee, etc. The perception can be expressed by soft factors like company image as well as customer satisfaction, product-service-innovation and number of returned customers. Finally business value is understandable as an integrative parameter, expressing the relationship between strategy, organizational performance and ICT via hard factors (e.g. financial power, turnover, etc.) and soft factors (e.g. market position, image, etc.). [14, 15]

3 Working with a multi-dimensional business value model

In a lot of software development projects requirements are prioritized with one simple business value expression (e.g. business value points in SCRUM). In our view this doesn't lead to a coherent and an overall understanding of the relevant strategy and goals. Due to that we will split the business value into several dimensions named domain values that mean that one business value is the result of several domain values. This domain values are focusing the three hierarchical domains "Strategy", "Tactic" and "Operation" of a company. Every domain is having a special focus to the company, its mission, its goals and its stakeholders and using a typical task-spectrum, influenced by quality dimensions as well as ideals and motivation (see Fig. 1). Therefore and in nature to Fig. 1 there is an own understanding of value and priority for example of a software development project on every domain.

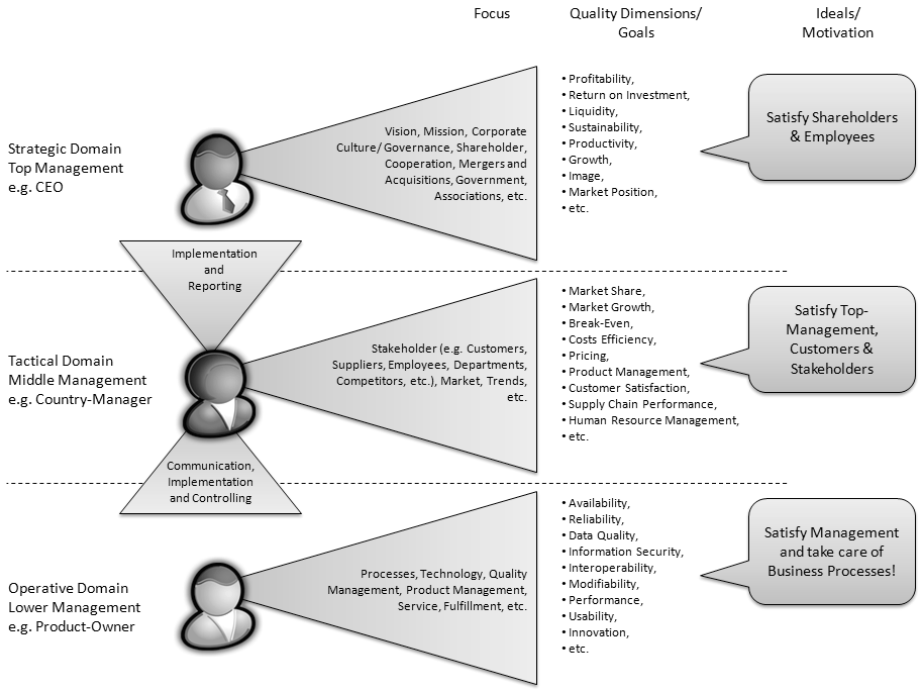


Fig. 1. Focus, Quality Dimensions and Ideals of the domains Strategy, Tactic and Operation

Among other contributions, this model seeks to allow (i) a better prioritization regarding to other domains, e.g. in agile SD-Projects, (ii) an enhanced focus about strategic goals and developments, (iii) a better understanding of market needs (especially for technical employees) and (iv) a strategic/value-control- and a strategic/value-feedback-system.

With a widespread view over all important business fields, this model can lead to a better business/strategy-orientation in agile software- and process-development of technology-driven companies in fast paced markets. It can offer the possibility of more specific requirements specification and prioritization as well as a better governance of daily business and software development projects. With our model, we aim to bridge the existing gap between business strategy and processes/ software. Tasks in the (development) processes are planned (i) in a timeline, (ii) in priorities according to the interests of the different business domains (hierarchical levels) / market views / technical views (iii) and results / increments are better traceable/checkable by every domain. Communication between stakeholders in projects based on the proposed BV-Framework is based on common tools, and is more efficient as conflicts and dependencies are quickly discovered and their influence on the overall outcome of the project can be clarified. It also ensures a better alignment to strategy and it is a tool for the evaluation of the contribution of individual sub-processes and software elements to the overall BVs.

4 Conclusion & Future work

This work introduces a multi-dimensional business value concept based on literature reviews and interviews with project managers as well as with experts beyond the theory. It is pointing to the problems in PM of technology oriented business models and leads to a possible solution as well as to a first discussion point with the community. This business value model is issue of further research. There is no general formula for a measurement of business value over domains and there is no structure to show ideals/ motivations. This challenge will be dealt with in further research. Based on this approach and with efforts to apply it to more use cases, literature reviews and feedback from scientists and practitioners, we'll try to increase the practical maturity as well as a scientifically validation. In addition to that and when this (agile) management approach based on the concept of domain values is ready for practical use we will link it and possible ways of value determination with the concept of CSFs especially CSFs for e/mCommerce (e.g. several trust building mechanisms). This means carrying the approach of the domain value model and CSF-based modeling of strategies into a software solution for early requirement engineering e.g. in the business engineering area and in agile project management.

5 References

1. Miniwatts Marketing Group: Internet Usage Statistics; http://www.internetworldstats.com/stats.htm; Access date: 19.04.2010
2. BITKOM: E-Commerce - Unternehmen; http://www.bitkom.org/de/markt_statistik/46259_38539.aspx; Access date: 10.11.2009
3. BITKOM: Online-Shopping in Deutschland weit verbreitet; http://www.bitkom.org/de/presse/30739_59337.aspx (Access date: 10.11.2009)

4. Bundesverband des Deutschen Versandhandels e.V.: http://www.versandhandel.org/News.80+M5c0e9546ce4.0.html; Access date: 16.11.2009
5. BITKOM: Mehr als 100 Millionen Mobilfunkanschlüsse in Deutschland; http://www.bitkom.org/de/presse/56204_51915.aspx; Access date: 10.11.2009
6. Statistisches Bundesamt: Bevölkerungsstand; http://www.destatis.de/jetspeed/portal/cms/Sites/destatis/Internet/DE/Navigation/Statistiken/Bevoelkerung/Bevoelkerungsstand/Bevoelkerungsstand.psml; Access date: 10.11.2009
7. E-Commerce-Center Handel: Geschäftsklima im E-Commerce 2009/2010; http://www.ecc-handel.de/download/83045001/Expertenstudie-Gesch%C3%A4ftsklima+im+E-Commerce.pdf; Access date: 16.11.2009
8. Rusnjak, A.; El Kharbili, M.: On Leveraging Business Processes to deal with Critical Success Factors. Workshop on Business Process Modeling and Realization, Informatik 2009, Luebeck, Germany, 2009
9. McLaughlin, Stephen: The imperatives of e-business: case study of a failed project; Journal Of Business Strategy Vol. 30 No. 1 (2009), Emerald Group Publishing Limited; www.emeraldinsight.com/10.1108/02756660910926966; Access date: 07.09.2009
10. Trigonum GmbH: Projektmanagement in verteilten Projektteams ; http://www.trigonum.de/fileadmin/trigonum.de/Dokumente/Studie%20-%20Projektmanagement%20in%20verteilten%20Projektteams.pdf; Access date: 17.11.2009
11. PA Consulting Group: Erfolgreich Projekte durchführen; http://www.gpm-ipma.de/php/fdownload.php?download=Studie_PA_und_GPM.pdf; Access date: 17.11.2009
12. Lee, Sungjae; Kim Kyoung-jae: Factors affecting the implementation success of Internet-based information systems; Elsevier Ltd.; http://dx.doi.org/10.1016/j.chb.2005.12.001; Access date: 18.10.2009
13. Sung, Tae Kyung; Gibson, David V.: Critical Success Factors for Business Reengineering and Corporate Performance: The Case of Korean Corporations; Elsevier Science Inc., 1998; http://dx.doi.org/10.1016/S0040-1625(98)00027-4; Access date: 18.10.2009
14. Matts, Chris; Pols, Andy: Business Value Driven Software Development; http://cdn.pols.co.uk/papers/businessvaluedrivendevelopment.pdf; Access date: 17.09.2009
15. Mahmood M.-A.; Gemoets, L.; Hall, L.-L.; Lopez, F.-J.; Mariadas, R.: Measuring E-Commerce Technology Enabled Business Value: An Exploratory Research; International Journal of E-Business Research, Vol. 4, Issue 2, IGI Global; http://www.infosci-journals.com/downloadPDF/pdf/ITJ4209_ICYdW2bbcf.pdf; Access date: 07.09.2009
16. Williams, Steve; Williams, Nancy: The Business Value of Business Intelligence, 2003; http://www.decisionpath.com/docs_downloads/BIJarticle.pdf; Access date: 17.09.2009
17. Tosic, Vladimir; Suleiman, Basem; Babar, Abdul: Specification of Business Value with and in Software Patterns; 2007; http://patterns-wg.fuka.info.waseda.ac.jp/SPAQU/proceedings/20-TosicSuleimanBabar-SPAQu07-Final.pdf; Access date: 18.09.2009
18. Agile Alliance: The Impact of Agile Processes on Requirements Engineering; http://www.agilealliance.org/system/article/file/906/file.pdf; Access date: 18.11.2009
19. Porter, Michael E. (2001): Strategy and the Internet; in: Harvard Business Review

6 First Workshop on Requirements Engineering in Small Companies (RESC)

Editors

Simone Bürsner
Bonn-Rhine-Sieg University of Applied Sciences, simone.buersner@h-brs.de

Thorsten Merten
Bonn-Rhine-Sieg University of Applied Sciences, thorsten.merten@h-brs.de

Technical Programme

RESC 2010: 1st Workshop on Requirements Engineering in Small Companies

Simone Bürsner and Thorsten Merten

Bonn-Rhine-Sieg University of Applied Sciences, Department of Computer Science,
Grantham-Allee 20, 53757 Sankt Augustin, Germany
{simone.buersner, thorsten.merten}@h-brs.de

1 Motivation and Goals

In the past RE research targeted mainly the needs of RE practice in the context of larger enterprises. However, Small and Medium Enterprises (SME) develop, customize and maintain a considerable part of software. Often, these companies are unable to apply RE methods and techniques without modifications. Besides, shortcomings in applying RE methods due to time constraints or limited resources may arise.

RE research has to intensify the investigation of RE practices in SMEs. Otherwise SMEs will have to continue their search for methodical orientation and dedicated tool support. Normally, the people responsible for requirements in SMEs are ambitious, but suffer from scarcity of resources. Their time for doing experiments and trying different methods is very limited. They need quick methodical improvement of requirements elicitation, documentation, communication and traceability as well as more continuity of requirements management through the whole software lifecycle. The practiced RE has to adopt the flexibility which is often an essential part of SMEs businesses.

The RESC workshop is intended to bring researchers together with the RE practices and experience of SMEs whose businesses are software.

2 Workshop Organization

The RESC workshop has been organized in the context of the research project KoREM[1] funded by the BMBF (Federal Ministry of Education and Research). In Germany, the BMBF supports research and development (R&D) projects at universities of applied sciences, whereas this support is connected to the cooperation with software SMEs because of their commercial relevance[2]. In the project KoREM, the Bonn-Rhine-Sieg University researches the connectivity of variants of existing RE techniques. This research targets towards lightweight RE methodologies and tool support usable for small companies.

[1] http://www.korem.de
[2] cf. http://www.bmbf.de/en/10785.php or http://www.bmbf.de/de/7152.php (German)

Organizational Committee:

- Simone Bürsner, (Bonn-Rhine-Sieg University of Applied Sciences, St. Augustin, Germany)
- Thorsten Merten, (Bonn-Rhine-Sieg University of Applied Sciences, St. Augustin, Germany)
- Barbara Paech, (Heidelberg University, Germany)
- Jörg Dörr, (Fraunhofer IESE, Kaiserslautern, Germany)

Program Committee:

- Jorge Aranda (University of Toronto, Canada)
- Simone Bürsner (Bonn-Rhine-Sieg University of Applied Sciences, St. Augustin, Germany)
- Jörg Dörr (Fraunhofer IESE, Kaiserslautern, Germany)
- Michael Ehresmann (insiders technologies, Kaiserslautern, Germany)
- Eva Geisberger (Technische Universität München, Germany)
- Andrea Herrmann (Technische Universität Braunschweig, Germany)
- Stefan Krause (DCON Software & Service AG, Kaiserslautern, Germany)
- Thorsten Merten (Bonn-Rhine-Sieg University of Applied Sciences, St. Augustin, Germany)
- Barbara Paech (Heidelberg University, Germany)
- Gerhard Pews (Capgemini sd&m AG, Frankfurt, Germany)
- Ernst Sikora (Duisburg-Essen University, Germany)
- Andreas Wachowski (XING AG, Hamburg, Germany)

3 Workshop Participants

The workshop participants have been mostly researchers as well as practitioners being interested in finding more insights regarding how small companies do RE currently as well as finding initial improvements for their RE.

Furthermore, ideas have been discussed on how to make a transfer of the good tacit knowledge usage in the loose organizational structures of small companies into larger companies.

4 Summary of Presentations and Discussions

On the Requirements Engineering in Small Companies (RESC) Workshop four paper presentations were given and discussed. Additionally, workshop chairs opened and concluded the workshop with corresponding presentations. Simone Bürsner gave an introductory presentation referring to the motivation of the workshop topic, the resulting workshop goals and the agenda. After the four presentations and associated discussions have been held, Thorsten Merten concluded the workshop by summarizing the content and introducing the final open discussion.

The Requirements Engineering in Small Companies (RESC) Workshop intended to show that small companies have specific RE needs. Naturally, the workshop asked for research questions as well as specific solutions for these companies and their RE.

At the RESC workshop the four papers imprinted below have been presented and discussed, whereas *the first paper* explored how different small companies work, *the second paper* explored ways research needs to go and *the other two papers* focused on specific solutions for ISO standardization as well as supporting the bidding phase.

The workshop and the discussions showed that RESC science is an important field that is not yet explored. Additionally, it found that existing RE techniques are not sufficient for small companies and that researchers need to get a better understanding of small companies, including the knowledge

- that size is an important, but not the only measure to categorize smaller companies and describe the exact focus of research,
- that tacit knowledge and social structures play an important role and that research may learn from functioning SMEs. This may lead to new methods making use of these circumstances,
- furthermore, these circumstances lead to natural advantages, which may be destroyed by introducing processes or RE methods, created for their larger counterparts without modifications,
- that methodologies still are not lightweight enough for small companies to be easily applied.

5 Acknowledgements

We would like to thank the workshop presenters and participants for their insights and their positive comments. We are pleased that all the participants argued for continuing the workshop at RefsQ in 2011.

Additionally, we would like to thank the program committee. We are also deeply grateful to the RefsQ-Organizers Ernst Sikora and colleagues for hosting RESC at RefsQ'10 in Essen.

The Benefit of Being Small: Exploring Market-Driven Requirements Engineering Practices in Five Organizations

Sami Jantunen,

Technology Business Research Center, Lappeenranta University of Technology
P.O. Box 20, FI-53851 Lappeenranta, Finland
sami.jantunen@lut.fi

Abstract. This paper explores five small and medium-sized organizations in order to understand how they manage their software product requirements. The paper illustrates the central role of human collaboration in small organizations and the challenges an organization meets when it begins to grow and lose opportunities for face-to-face collaboration. The findings of this study suggest that organizations need to find new ways to gain the benefits of human collaboration while coping with increased complexity. The paper calls for requirements engineering approaches that help people to cope with multiple meanings, are more social by nature and are more tolerant of improvisation than the traditional designs grounded in decision making. Studying small organizations and investigating the opportunities with social media are two potential sources in such line of research.

Keywords: Market-Driven Requirements Engineering, Human Factors, SME

1 Introduction

One of the most crucial tasks for a company offering a software product is to decide what new features shall be implemented to the product's forthcoming versions [1], [2]. This task is usually considered as the concern of Market-Driven Requirements Engineering (MDRE) [3], which adds the marketing perspective to RE [2]. In general, the MDRE processes have been described as approaches to synchronize the continuous flow of candidate requirements and the work with the discrete release events [3], [4].

Many companies have become increasingly overwhelmed with the complexity of their design problem aiming at determining the best possible selection of features for the product's future releases. This problem has been claimed to be wicked [2]. There are better or worse solutions for the problem, but no optimal one [2]. The criteria determining the success of the solution keep changing due to continuously changing competition, technologies and market needs. To make matters worse, the complexity of the problem tends to increase over time. New customers may introduce new requests that may conflict with the needs of previous customers [5].

While the processes and related challenges of managing product requirements are well known for larger organizations [1], [2], [6], [7], small software companies have been claimed to manage their requirements in ways that bear no relation to what the textbooks say, and what is taught in undergraduate courses [8]. Hence, this paper explores five small and medium-sized software development organizations in order to understand better how organizations manage their product-related requirements in practice. In particular, the study seeks to answer how product development-related challenges and practices can be characterized in small software product development organizations compared to larger ones.

The remaining part of this document is organized as follows. Section 2 describes the research process and methods used in this study. Section 3 presents a conceptual framework that emerged from the data analysis, describing how organizations manage their product requirements. This framework is then utilized in Section 4 to illustrate two extremes in a continuum of managing product requirements. Section 5 discusses the findings and, finally, conclusions are drawn in Section 6.

2 Research Methodology

The data for this study has been gathered in conjunction with a large research project aiming at supporting the internationalization of knowledge-intensive companies[1]. The primary data collection method for the project has been theme-based interviews that were in most cases conducted by two out of the four project's researchers. The themes in these interviews included internationalization, partnerships, business strategies and product development collaboration. Altogether the research project has produced close to 100 transcribed interviews from 40 software development organizations. In addition to the interviews, we have gathered documents revealing details of companies' software development practices. Based on the organization's focus on software product development and the richness of the available product development-related empirical data, five companies participating in the research project were selected for further analysis, reducing the number of transcripts to be analyzed into 34 interviews (Table 1).

The analysis was conducted using Atlas.ti [9], a software specifically intended for qualitative data. Following the early steps of a grounded theory approach [10], the data has been analyzed line by line, constantly coding each sentence and by thinking of multiple possible interpretations and assigning more than one code to a segment of data when applicable. The line by line approach has been claimed to force the analyst to verify and saturate categories, minimize the chance of missing an important category, produce a dense rich theory and give a feeling that nothing has been left out [11]. Furthermore, focusing on small portions of the data at a time helps to ensure that none of the analyst's "pet themes" will be incorporated into the theory unless they have an emergent fit with the data [11].

[1] For more information, please visit Global Network Management project's website: http://www.tbrc.fi/gnm/.

Table 1. Case companies

= Company number, S = Size of product organization, I = Interviews conducted

#	Product offering	Product's business environment	S	I
1	2D and 3D CAD-design and data management solutions	• Matured product domain. • Some customers and partners also in neighboring market areas.	<25	5
2	Solutions for managing resources and material flows	• Requires customization for each customer • Customer delivery projects tend to take resources from product development. • Largely domestic customers	<25	10
3	RFID –based identification solutions.	• Embedded systems. Some customization required. • Matured product domain. • Some customers and partners also in neighboring market areas.	~25-50	7
4	Model-based software product for building and construction	• Diverse customer needs. • Complex and changing product domain. • Customers in over 80 countries; partners in close to 30 countries; own country offices in close to 15 countries.	>300	6
5	Engineering and data management software	• Several productized solutions. Some customization required. • Customers in more than 30 countries; some own country offices and foreign partners.	<100	6

After the essential sections of gathered data were assigned with conceptual codes, the analysis proceeded to the next phase in which the relationships between the identified codes were in focus. Utilizing again functionality of Atlas.ti [9], the codes from all interviews of a particular organization were exported to one organization-specific network diagram, where each of the conceptual codes were represented as individual boxes. The analysis proceeded from here by visually organizing and connecting conceptual codes until the diagrams formed maps of clearly definable interconnected clusters with similar codes next to each other. These organization-specific maps revealed similarities and differences between the companies in terms of software product development related challenges and practices. This paper reports findings that resulted from the comparison of the organization-specific diagrams.

3 MDRE in Practice: Emerged Conceptual Description

In this section, the commonalities among the analyzed organizations are developed further into a conceptual description of MDRE in practice. The conceptualization begins with the recognition of tacit knowledge as an important factor affecting the product related decision making. Since tacit knowledge is deeply rooted in action, commitment, and involvement in a specific context [12], the data analysis first identified three contexts within people participate when determining what functionality shall be implemented to the product's forthcoming versions. These

contexts included: 1) *sensing the market* for emerging trends and the needs of customers as well as transmitting the gathered information to people responsible for managing the product; 2) *making sense of the market* by sorting the gathered needs, resolving conflicts and inconsistencies, prioritizing the requirements and planning for the implementation; and 3) *acting upon knowledge* by communicating the future plans to relevant stakeholders and determining the suitability, reasonableness, consistency, completeness, and lack of defects in a set of requirements.

People participating within these three contexts appeared to go through similar activities in all of the analyzed organizations. These identified activities included: 1) *listening* for information about emerging trends and customers' needs; 2) *sharing* the gathered product-related needs with others; 3) *determining* what the actual requirement really is. A common challenge in organizations was the fact that customers are often unable to articulate what they really need; 4) *knowing* the impact of the proposed requirement. This activity includes building the understanding of the requirement's value from different viewpoints and determining requirement's conflicts and dependencies with other requirements; 5) *deciding* which of the requirements shall be implemented; 6) *understanding* and communicating the made decision to the ones responsible to act on them; and 7) *implementing* the decisions made.

The three contexts people are involved in and the seven activities of determining what shall be implemented into product's forthcoming versions create a framework for further analysis (Fig. 1).

Fig. 1. Three contexts and seven activities of determining what shall be implemented into forthcoming product versions.

4 Different MDRE Approaches

The data analysis revealed two extremes in a continuum to conduct MDRE activities. While companies #1 and #2 (Table 1) based their software product development practices on *collaboration*, the emphasis of product development activities within the remaining three case companies was on following *processes*. Although size of an organization seemed to be a major factor affecting the organization's software development style, it was recognized that the number of employees is not the only factor to be considered. Other factors affecting the software

development style have been reported to be: type of the customers, background and skills of the developers, the preferences of the company's founders, nature of the business environment, and the spatial layout and geographical distance of the offices [8]. Hence, a decision was made to avoid creating labels suggesting classifying organizations based on their headcount. Instead, the two identified approaches to manage product requirements were labeled as *collaboration-based MDRE approach* and *processes-based MDRE approach*. These approaches are illustrated further as follows.

4.1 Collaboration-based MDRE Approach

All organizations basing their software development style on collaboration had only a handful of people working with the product. Characteristic to such organizations was that there are no clear roles. Everyone does many different kinds of tasks:

"The benefit of working in a small company is that when you wear a tie instead of t-shirt, you know that you are currently doing sales work."
-software developer in a collaboration-based company

As long as the organization is small enough, work was conducted rather informally, heavily relying on collaboration:

"The two of us hold a face-to-face meeting in which we go through what we need to do for the next product version. Then, we present the roadmap to our management who ask reasons for our decisions."
-two resources responsible of a product in a collaboration-based company

"We have such a light organization. We can sort things out while having a cigarette."
-two resources responsible of a product in a collaboration-based company

Relying largely on human collaboration was typically positively experienced:

"I don't necessary long for a fancy organization that produces loads of documents. I'd rather do real work."
-software developer in a collaboration-based company

Informal collaboration-based MDRE appeared to have clear benefits. The fact that people perform diverse set of tasks enabled them to utilize tacit knowledge they have gained in their day-to-day social interaction. Since tacit knowledge is difficult to formalize and often time- and space-specific, tacit knowledge can be acquired only through shared direct experience, such as spending time together or living in the same environment [13]. Furthermore, continuous face-to-face connections allow workers to share information with means much more powerful than with documents. Hence, human collaboration appeared to be a significant factor in mitigating the challenges when conducting activities of *listening, sharing, determining, and knowing* the product requirements. However, informal practices may also have unwanted consequences. For example, the lack of formal practices may lead to individualism:

"We all have long traditions of working our own individual ways. If you would try to have all of us work in a uniform way, you would fail."
-software developer in a collaboration-based company

"He has his own way of thinking on certain matters and he implements his thoughts without discussing with others. When others disagree, there may be some shouting on the hallways.

There are quite many propellerheads here with a strong personality."
-project manager in a collaboration-based company

Informality and individualism also introduces risks:
"If I get hit by a tram on my way to home, that's about it. The product-related knowledge would be pretty much lost."
-software developer in a collaboration-based company

Strong personalities tend to dominate decision making in small organizations. In some companies, the product development decisions were made by the developers:
"If the sales guys would get even a slightest hint of what we are playing around with, they would immediately call to the customers and use the information to sell a feature before we have decided to implement it. That's why we are very cautious of revealing information to the sales guys."
-software developer in a collaboration-based company

In other companies, decisions were made by sales:
"I'd say that 80 percent of the decisions to implement a new feature have actually been made by selling the non-existing functionality to a customer."
-salesman in a collaboration-based company

Hence, relying on human collaboration may introduce risks particularly for activities of *deciding*, *understanding* and *implementing* the requirements. Furthermore, informal practices appear to work only up to certain point. As the organization begins to grow, the need for more formal work practices increases:
"At the moment I do all the coding. If we would hire even one more person, things would get 10 times more complicated. We would need to start synchronizing and agreeing on everything and start assigning responsibilities."
-software developer in a collaboration-based company

The first symptoms for the need of more formal practices appear to be coordination problems
"We have too many competing versions that we have implemented to different customer projects. Other project may start from scratch implementing a solution that we have already implemented for another project. The developer may not have heard about the existing solution or he prefers to develop the solution himself rather than use the existing one."
-project manager in a collaboration-based company

and version control problems:
"Occasionally someone comes to notify me that he has just saved a new version of a source file I was also working on. The fact that my recent work was lost makes me often annoyed enough to overwrite his work by saving my version of the file."
-project manager in a collaboration-based company

4.2 Process-based MDRE Approach

As the organization grows, its product-related customer and user needs begin to originate from a wide variety of sources:
"Some of the sources through which we receive market information, in one way or another, are: the existing customers, the ongoing sales cases and all sorts of ideas we get at the exhibitions while looking at what competitors have accomplished. We also follow, to some extent, what is happening in the adjacent customer segments."
-head of product management in a process-based company

The people who gather the product related needs are not necessarily any more the same compared to the ones who use the information to make decisions. Typically, product-related needs are collected on a continuous basis and are stored into a database. Such way of working did not appear to introduce notable challenges for activities of *listening* and *sharing* the requirements (Fig. 1). The companies tended to gather plenty of market information – perhaps more than they were able to digest:

”We receive large amounts of market information, but the typical problem we are facing with it is that the business value behind the customer need is often missing. In such case, we have difficulties on prioritization. We might not be able to see that focusing on other requests would actually benefit us much more. We have a horn of plenty on receiving market information, but understanding the priority of information often gets lost in the abundance of technical details.”
-business manager in a process-based company

The real challenge appeared to be to understand what the gathered information really means. The database was typically periodically scanned in order to decide which of the requirements shall be implemented for the product's forthcoming releases. Essential processes when making such decisions are *requirements prioritization*, *roadmapping* and *release planning* [3]. Artifacts that are produced when following such processes are: *a roadmap* that provides a layout of the product releases to come over a time frame of three to five years and *a release plan* that describes the selected features to be implemented for the particular product release.

The gathered information needed first to be developed further into explicit product requirements. It was then necessary to develop an understanding of how important the product requirements were in terms of product business. Without such understanding it was difficult to determine which of the requirements should be implemented to the product's forthcoming versions. These tasks were typically assigned to a product management team:

“One of our product management team's responsibilities is to develop an understanding of what features would be technologically viable and desirable for the customers. The product management team need to see beyond the customer wishes and think of functionality that the customers are not yet able to request for.”
-business manager in a process-based company

Building an understanding of the gathered information appeared to be a complex challenge, for which decision makers' experience and tacit knowledge appeared to be important:

“There does not exist any equation that can determine the priorities of market needs correctly. It takes certain touch, hunch and experience to understand the priorities. This knowledge has just been built into the organization. [...] The more we have made business, the more we have gained this tacit knowledge.”
-business manager in a process-based company

"It is a huge challenge to set up the organization to support the product-related decision making. It is difficult to determine the criteria according which the requested features are decided to be implemented. We have recently put much effort in finding ways to organize the decision making in a business oriented manner.”
-business manager in a process-based company

“Who is it to say that what is the right interpretation of the data we have gathered? This is the challenge that we have been tackling with all the time. The more we have done business the

more experience we have gained [on understanding the customers]"
-business manager in a process-based company

In addition to the challenge related to understanding and decision-making, another challenge was clearly identified. There appeared to be communication blockages between the product management team and the rest of the organization:

"We have detected challenges on how to share the information between the customer segment teams and the product management team. In addition, we have identified another communication barrier between the product management team and the development team. If we can solve these two Gordian knots then everyone's work will be easier."
-business manager in a process-based company

Not knowing the full picture of product-related decisions have occasionally caused challenges in the organizations:

"In many cases, knowing the plans for the future versions would have an impact on the design decisions. If we would know that a certain requirement is actually laying a foundation to something forthcoming, we would implement the requirement differently."
-software development team leader in a process-based company

"We have had challenges on informing the customers about the practical meaning of new product features. We have listed what features the new product version contains, but the true practical meaning has not been understood."
-head of product management in a process-based company

"Our marketing department has not been able to write anything related to the new product until the product has been implemented."
-head of product management in a process-based company

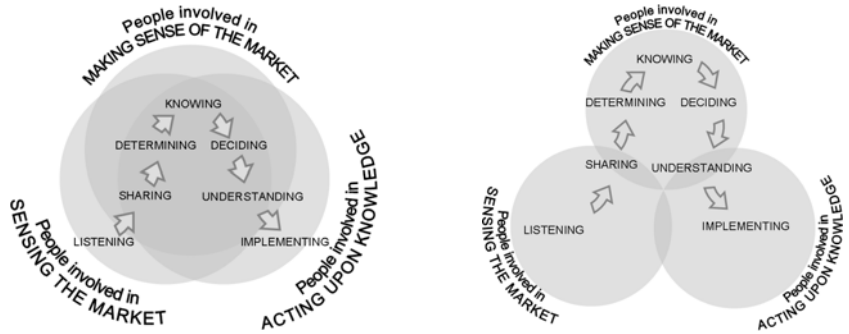
"The R&D department has only been able to see a planning window of one third of a year. Because of this, they do not know what features are to be implemented on the next version."
-head of product management in a process-based company

"The country managers have had to rely on product manager's communication skills in creating an understanding of what is new in the products forthcoming versions."
-head of product management in a process-based company

The gathered data suggested that process-based companies communicated largely with the artifacts resulting from following processes. This appeared to be a source of misunderstandings. The organizations' movement from collaboration towards following processes have created challenges particularly to activities of *determining*, *knowing*, *deciding*, *understanding* and *implementing* the requirements.

5 Discussion

The conceptual summary characterizing collaboration- and process-based organizations is presented in Fig. 2. The findings of this study support the claim [14] that co-operation and frequent face-to-face communication have a central role within the smaller development teams. As long as an organization is small enough, human collaboration appears to have a natural tendency to mitigate the MDRE-related challenges. When the organization grows, it begins to face coordination challenges.



Collaboration-based organizations **Process-based organizations**

Fig. 2. Conceptualized differences between collaboration- and process-based organizations.

Organizations typically alleviate such challenges by introducing processes. With processes, people begin to specialize and information begins to be shared more in documented form. Paradoxically, with the efforts to increase coordination, organizations appear to lose the benefits of human collaboration. A growing organization may soon find itself in the middle of communication challenges where people need to work with partial information. With the reduced human collaboration, growing organizations destroy the very thing that once made them perform well.

The findings of this study suggest that organizations need to find new ways to regain the benefits of human collaboration while coping with increased complexity. In particular, current MDRE approaches could be expanded with approaches that help people to cope with multiple meanings, are more social by nature and are more tolerant of improvisation than the traditional designs grounded in decision making. Studying how small organizations develop their requirements may be a valuable source of information in accomplishing such a goal.

The emergence of social media [15] has created high expectations for their opportunities to alleviate many problems related to social interactions over time and place. Some of the essential characteristics of social media include support for social interaction, content sharing, virtual identity and collaborative production [16]. These characteristics may also provide value in understanding complex and often conflicting product-related needs, because "the greater the variety of beliefs in a repertoire, the more fully should any situation be seen, the more solutions that should be identified, and the more likely it should be that someone knows a great deal about what is happening" [17].

6 Conclusions

This paper explored five organizations in order to understand how they manage their software product requirements. The study illustrated how small organizations tend to manage their product requirements largely with human collaboration whereas larger organizations relied on processes. Human collaboration appeared to have a natural tendency to mitigate MDRE-related challenges. Hence, this study argues that studying

small organizations and investigating the opportunities with social media are two important sources of information when finding new ways to gain the benefits of human collaboration while coping with increased complexity. The question still remains, what concrete actions organizations should take to release these new potentials in requirements engineering. The challenge for product organizations is how to utilize the wisdom of its organization in order to determine the product related need while producing a simple enough understanding for the ones needing to act on it.

References

1. D. E. Damian, D. Zowghi: RE challenges in multi-site software development organisations. Requirements Engineering. 8. 149-160 (2003)
2. P. Carlshamre: Release Planning in Market-Driven Software Product Development: Provoking an Understanding. Requirements Engineering. 7. 139-151 (2002)
3. B. Regnell, S. Brinkkemper: Market-Driven Requirements Engineering for Software Products. In: A. Aurum and C. Wohlin (eds.) Engineering and Managing Software Requirements. pp. Springer. (2005)
4. I. van de Weerd, S. Brinkkemper, R. Nieuwenhuis, J. Versendaal, L. Bijlsma: Towards a Reference Framework for Software Product Management. In: 14th IEEE International Requirements Engineering Conference, pp. 319-322. Minneapolis/St. Paul, Minnesota, USA (2006)
5. S. Jantunen, K. Smolander, D. C. Gause: How Internationalization of a Product Changes Requirements Engineering Activities: An Exploratory Study. In: 15th IEEE International Requirements Engineering Conference (RE 2007), pp. 163-172. New Delhi , India (2007)
6. L. Lehtola, M. Kauppinen: Suitability of Requirements Prioritization Methods for Market-driven Software Product Development. Software Process Improvement and Practice. 11. 7-19 (2006)
7. L. Karlsson, Å. G. Dahlstedt, B. Regnell, J. Natt och Dag, A. Persson: Requirements engineering challenges in market-driven software development -An interview study with practitioners. Information and Software Technology. 49. 588-604 (2007)
8. J. Aranda, S. Easterbrook, G. Wilson: Requirements in the wild: How small companies do it. In: 15th IEEE International Requirements Engineering Conference, pp. New Delhi, India (2007)
9. Scientific Software Developmen GmbHt: Atlas.ti. (2004)
10. B. Glaser, A. L. Strauss: The Discovery of Grounded Theory: Strategies for Qualitative Research. Aldine. Chigago, (1967)
11. B. G. Glaser: Theoretical Sensitivity. The Sociology Press. Mill Valley, (1978)
12. I. Nonaka: A Dynamic Theory of Organizational Knowledge Creation. Organization Science. 5. 1, 14-37 (1994)
13. I. Nonaka, R. Toyama: The knowledge-creating theory revisited: knowledge creation as a synthesizing process. Knowledge Management Research & Practice. 1. 2-10 (2003)
14. C. Hansson, Y. Dittrich, B. Gustafsson, S. Zarnak: How agile are industrial software development practices? Journal of Systems and Software. 79. 1295-1311 (2006)
15. Y. Benkler: The wealth of networks: how social production transforms markets and freedom. Yale University Press. New Haven, (2006)
16. K. Lietsala, E. Sirkkunen: Social Media - Introduction to the tools and processes of participatory economy. (2008)
17. K. E. Weick: Sensemaking in Organizations. Sage Publications. Thousand Oaks. California, (1995)

Playing to the strengths of small organizations

Jorge Aranda

University of Toronto,
Toronto, ON, Canada
jaranda@cs.toronto.edu

Abstract. By virtue of their size, small organizations can take advantage of many opportunities to develop software efficiently and successfully, and they waste them if they try to emulate their larger counterparts. As software researchers, we should study how small organizations can best exploit those opportunities instead of prescribing solutions that were designed for organizations of a very different nature.

"Today, we suffer from an almost universal idolatry of gigantism. It is therefore necessary to insist on the virtues of smallness, where this applies." —E. F. Schumacher [16]

1 Introduction

Small organizations form a large part of the software industry,[1] but our research community has mostly overlooked their needs and characteristics. This is a serious omission: small software organizations have a number of strengths that are absent in larger organizations, strengths that help them develop software efficiently and successfully and that are neglected if the organization applies processes and practices that were not designed to exploit them.

In this paper I argue that small organizations should not attempt to emulate the processes and practices of larger organizations, but should rather take advantage of the strengths enabled by their size. Similarly, I argue that requirements researchers should recognize these strengths, and design and evaluate techniques that make the best use of them.

2 Small is beautiful

We have known since the earliest days of our field that large software organizations suffer from problems caused by their size. They incur in significant coordination overheads [6], and tend to release products that are less satisfactory than those built by smaller organizations [9]. But somehow many in the

[1] In the United States in 2002, 95% of software development firms had less than 50 employees. They generated 21% of the total income and employed 28% of all employees in the area [7].

software industry assume that the goal of a small firm should be growth, that size is a valid measure of success. To be sure, a large size brings certain benefits: the appearance of stability, the ability to engage in greater and more ambitious projects, the appeal of commanding the work of a large number of employees. And yet there are many rewards for small organizations, rewards that often go unnoticed and unclaimed in their push to become large by behaving as if they were already large. Some of these rewards are psychological and even ethical, such as the joy of working in closely-knit groups and a greater agency over one's own work. That kind of reward may be significant enough to justify a preference for small groups, but it is not the topic of this paper. Rather, I claim that small organizations also have important advantages purely from the point of view of developing software efficiently and successfully. Some of these advantages relate specifically to their requirements elicitation and communication activities.

2.1 Formality is unnecessary

Organizational scientists tell us that increases in organizational size lead to greater bureaucracy and formalization [5, 12]. Large organizations succeed partly by being predictable; predictability is achieved through organizational inertia and the formalization of structures and processes [11]. Requirements must be elicited by specialized personnel, documented in formal and unambiguous terms, traceable back to their sources and forward to their implementations, and changed only with the oversight of a committee. Only through mechanisms such as these can large organizations deal with the challenge of communicating and controlling the requirements of their projects.

For small organizations, many of these activities are entirely unnecessary; there are plenty of documented cases of successful organizations that do fine without them [2]. This is largely because it is easier to share an understanding of requirements information with everyone involved on an as-needed basis. If a team is able to sort out its requirements problems by getting everyone together in the same room, it does not need to spend time creating documents that will soon become obsolete and might go unread [14].

2.2 Communication can be rich and robust

Organizations working on large projects must adopt some form of geographic distribution of effort. Even when the whole organization is located in the same area, communication between those of its members sitting beyond a short distance from each other is as low as if they were in different cities [1]. This is one of the factors that force large organizations to use inefficient communication mechanisms, such as requirements documents, to share project information.

In contrast, if the organization is small enough that it can work in a shared room or two, it is able to use much richer and pervasive communication dynamics [15]. This "radical co-location" has been found to lead to greater project efficiency and satisfaction [17], and it allows the organization to forego the creation and maintenance of unnecessary documentation. Note that the effects of

co-location benefit large organizations as well when they can partition projects in small sizes, but this is a natural advantage for small organizations.

2.3 Strong cohesion is possible

Small organizations can develop a strong group cohesion with relative ease. Group cohesion leads to increases in performance [3], partly because cohesive groups have lower coordination and communication overheads. They develop a shared vocabulary and a tacit understanding of each member's areas of expertise, enabling the maintenance of an efficient "oral tradition" within their teams.

This cohesion can even extend to members of customer organizations. Extreme Programming [4], for instance, advocates for the development of a close bond with customers, a bond that allows the team to understand the needs and culture of their clients intuitively and to resolve technical issues quickly.

2.4 Unscalable practices can be implemented

Many of the software development practices popularized in recent years, particularly those based on the Agile manifesto, prioritize co-located, cohesive teams over formalized processes. Arguably, part of the backlash against the Agile movement comes from the mismatch between its proposals and the formalized, geographically distributed, incohesive environments in which people attempt to apply them. It is possible to be Agile in large organizations, but it is not easy.

Agile techniques are a much better fit to smaller (less formal, co-located, cohesive) organizations. Story cards, backlogs, daily sprints, and other agile requirements practices depend on such an environment to prosper. They are tested, validated strategies to understand, prioritize, and track the requirements of a project, but they do not scale well—a problem for large, but not for small organizations.

3 Conclusion

We should not try to persuade small organizations to use the strategies we have devised for their larger counterparts. Some small organizations do apply them, perhaps out of a belief that it is the correct way to develop software, or a desire to emulate seemingly successful large firms. My position is that this is a mistake. Those strategies do not take advantage of their strengths; in fact they waste them entirely. Instead of proceeding down this path, we should welcome the opportunity to help these organizations identify their abilities, and to be explicit about the ways in which these abilities can be exploited to their advantage.

An important question remains: when does a growing organization cease to be small? That is, when do these strengths disappear? There seems to be a threshold after which organizational dynamics change, at around ten or twenty people [8]. Many organizations appear to have another qualitative jump at about one hundred and fifty [10]; there could be at least one more in-between these

two. To my knowledge, these thresholds have yet to be explored in the domain of software organizations. And it is possible that other determinants of size, such as the number of teams or the number and variety of customers, is more important than the number of employees for our purposes [13]. A better understanding of the construct of size and of the characteristics of software organizations of different sizes is an important step to advance our knowledge of the field.

4 Acknowledgements

I would like to thank Steve Easterbrook, Greg Wilson, Jon Pipitone, Neil Ernst, and Jonathan Lung for their insightful comments on this paper.

References

1. Thomas J. Allen. *Managing the Flow of Technology.* MIT Press, 1977.
2. Jorge Aranda, Steve M. Easterbrook, and Gregory V. Wilson. Requirements in the wild: How small companies do it. In *RE '07: Proceedings of the 15th IEEE International Requirements Engineering Conference*, pages 39–48, Delhi, India, 2007.
3. Daniel J. Beal, Robin R. Cohen, Michael J. Burke, and Christy L. McLendon. Cohesion and performance in groups: A meta-analytic clarification of construct relations. *Journal of Applied Psychology*, 88(6):989–1004, 2003.
4. Kent Beck. *Extreme Programming Explained: Embrace Change; 2nd Edition.* Addison-Wesley Professional, 2005.
5. Peter M. Blau and Richard A. Schoenherr. *The Structure of Organizations.* Basic Books, 1971.
6. Frederick P. Brooks. *The Mythical Man-Month.* Addison-Wesley, 1975.
7. U. S. Census Bureau. *Statistics retrieved from http://www.census.gov/.*
8. Theodore Caplow. Organizational size. *Administrative Science Quarterly*, 1(4):484–505, 1957.
9. Melvin E. Conway. How do committees invent? *Datamation*, 14(4):28–31, 1968.
10. Robin Dunbar. *Grooming, Gossip, and the Evolution of Language.* Harvard University Press, 1996.
11. Michael T. Hannan and John Freeman. *Organizational Ecology.* Harvard, 1989.
12. Heather A. Haveman. Organizational size and change: Diversification in the savings and loan industry after deregulation. *Administrative Science Quarterly*, 38:20–50, 1993.
13. John R. Kimberly. Organizational size and the structuralist perspective: A review, critique, and proposal. *Administrative Science Quarterly*, 21(4):571–597, 1976.
14. Timothy C. Lethbridge, Janice Singer, and Andrew Forward. How software engineers use documentation: The state of the practice. *IEEE Software*, 20(6):35–39, 2003.
15. Gary M. Olson and Judity S. Olson. Distance matters. *Human-Computer Interaction*, 15(2):139–178, 2000.
16. E. Fritz Schumacher. *Small is Beautiful: A study of economics as if people mattered.* Blond and Briggs, 1973.
17. Stephanie D. Teasley, Lisa A. Covi, M. S. Krishnan, and Judith S. Olson. Rapid software development through team collocation. *IEEE Transactions on Software Engineering*, 28(7):671–683, 2002.

ISO-Standardized Requirements Activities for Very Small Entities

Philippe Saliou and Vincent Ribaud

Université de Brest, LISyC, CS 93837, 29238 Brest Cedex, France
Université européenne de Bretagne, France
{ Philippe.Saliou@univ-brest.fr, Vincent.Ribaud@univ-brest.fr }

Abstract. The use of Software Engineering standards may promote recognized and valuable engineering practices for Very Small Entities (VSEs) but these standards do not fit the needs of VSEs. The ISO/IEC Working Group 24 (WG24) is developing the ISO/IEC 29110 standard "Lifecycle profiles for Very Small Entities"; this standard is due for approval in June 2010.
A pilot project about ISO 29110 use has been established between our Software Engineering group and a 14-person company building and selling counting systems about the frequentation levels of public and private sites. The pilot project aims to help VSEs deliver the Software Requirements Specification, Test Cases and Test Procedures for a new web-based system intended to manage fleets of counting systems. As the project goes along, it appears that the 29110 set of documents was not up to the task of sustaining this VSE in its engineering activities. We supported the VSE in two ways: (i) a Training Session based on the 29110 Requirements Analysis activity, and (ii) Self-Training Packages - a set of resources intended to develop experience and skills in Requirements Identification and SW Requirement Specification (SRS). Our inspiration stems from the 15504-5 standard with a desire to provide software engineers with an exemplar set of base practices providing a definition of the tasks and activities needed to fulfil the process (e.g. requirements) outcomes. Task definition is collected on a task card. The results of this pilot study provide the VSE with a roadmap through the Requirements activity, which is compatible with the ISO/IEC 29110 standard.

Keywords: Very Small Entities, Requirements Specification, ISO/IEC 29110.

1 Introduction

Very Small Entities (VSEs) are recognized as being very important to the software economy, and produce stand-alone or integrated software components in large software systems. The use of Software Engineering standards may promote recognized and valuable engineering practices - but these standards do not fit the characteristics of VSEs. The term 'Very Small Entity' (VSE) was defined by the ISO/IEC JTC1/SC7 Working Group 24 (WG24) as being "an entity (enterprise, organization, department or project) having up to 25 people". This definition has subsequently been adopted for use in the ISO response to VSEs' specific needs: the

emerging ISO/IEC 29110 standard "Lifecycle profiles for Very Small Entities" [1]. The 29110 standard defines a group of Standardized Profiles, including the ISO/IEC IS 29110-4-1 Basic profile [2] which applies more specifically to a VSE that is involved in software development of a single application by a single project team with no special risk or situational factors.

A VSE claiming compliance with ISO/IEC IS 29110-4-1 will implement and use all the profile elements, as identified in Clause 7 of the profile specification [2]. The profile elements concerning requirements are: Project Plan Execution (PM.2) and Project Assessment and Control (PM.3) - producing the *Change Request* work product, and Software Requirements Analysis (SI.2) - producing work products *Change Request* and *Requirement Specification*.

This paper reports some of the conclusions reached by a pilot project the authors conducted with a 14-person VSE that builds and sells counting systems about the frequentation of private and public sites. Only 3 of the employees are software developers, and the VSE asked for assistance with software project management – mainly managing requirements and establishing a disciplined test process. Deployment Packages (DP) are expected to be particularly helpful. A DP is "a set of artefacts developed to facilitate the implementation of a set of practices, for the selected framework, within a VSE [3]". As the project goes along, it appears that the 29110 set of documents (including DPs) was not up to the task of sustaining this VSE in its engineering activities. One idea defended here is that implementing standardized software engineering activities in a VSE requires specific and operational materials and mechanisms. We are proposing to provide VSE employees with Self-Training Packages intended to help the engineer carry out [and learn] the task.

Section 2 presents related work, and offers an overview of a SE standard for VSEs. Section 3 introduces the pilot project, presents Self-Training Packages, and evaluates the system's efficiency. We conclude with brief perspectives.

2 Related work

2.1 Requirements engineering for small software companies

In 2007, IEEE Software published a special issue on the theme "SE Challenges in Small Software Companies". The guest editors' introduction presents common challenges faced by large and small software development companies: "They need to manage and improve their software processes, deal with rapid technology advances, maintain their products, operate in a global software environment, and sustain their organizations through growth [4]". Yet VSEs also have specific characteristics and needs.

J. A. Calvo-Manzano et al. [5] presented an SPI solution called MESOPYME for small and medium-size enterprises (SME). MESOPYME is based on the Action Package concept - a mechanism which assists faster and affordable SPI program implementation for SMEs. Experimentation with this package has been carried out in the Requirements Engineering domain. The structure of an Action Package (such as

the Requirements Engineering Action Package) presents similarities to our own structure of Self-Training Packages. Training is provided using the Action Package Training component. This component basically comprises four courses: software process model (CMM), the improvement method (MESOPYME), team building, and training in the process selected for improvement (e.g. Requirements Engineering). Our approach is different in that MESOPYME is a Software Process Improvement method for SMEs, whereas we aim to implement a Lifecycle Standardized Profile in VSEs.

The REDEST project [6] aimed to develop a selection of innovative Requirements Engineering methodologies to act as Best Practice Cases for 14 independent software development companies. REDEST disseminated results via a Best Practice Case Booklet [7]. Case Study 8, carried out by a VSE named SignalKomplex, aimed to experiment with the following features: introduction of a systematic RE process; a more thorough understanding of customer requirements; basic tracking of changes in requirements. The size (24 employees) and the products and services (vehicle traffic control equipment) provided are very similar to the VSE case study reported in this paper. SignalKomplex baseline project (development of a vehicle sensor card) presents similarities with the VSE project (a web-based system intended to manage fleets of counting systems). The RE approach selected by SignalKomplex was a method called PAISLEY, which is an approach whose focus couples Requirements Elicitation with the processes of the object being developed. SignalKomplex selected this approach because it was equally operable for hardware and software requirements, a key issue from the SignalKomplex point of view. As SignalKomplex reported in the REDEST Best Practice Case Book [7, p. 114], the RE solution also required input from other areas of the company, such as the sales and business departments. Combining pure, technical specifics with other inputs was mostly achieved by exploiting spreadsheet features. The ISO/IEC 29110 Basic Profile is applicable to VSEs which do not develop critical software products, and the traceability tool provided with the Deployment Package associated with requirements is a spreadsheet-based tool. Our proposal is to perform a preliminary Requirements Elicitation through the building of a Services Identification List (see Figure 1) which is also supported by spreadsheets. Keeping a powerful requirements management tool as simple as possible is a key issue for a VSE.

2.2 SE Standards for Very Small Entities

ISO initiative. Software engineering standards and methods often neglect the needs and problems of the small and medium-sized organizations which constitute a major part of the software industry. The ISO/IEC Working Group 24 (WG24) is developing the emerging ISO 29110 standard, which is a set of technical specifications and guides for use by very small software enterprises. This set is based on the concept of VSE profile [1]. The purpose of a VSE profile is to define a subset of ISO/IEC standards relevant to the VSE context - for example, selected processes and outcomes of ISO/IEC 12207 [8] and selected products of ISO/IEC 15289 [9].

ISO/IEC 29110 Set of Documents. The ISO/IEC 29110 Set of Documents comprised multiple documents (overview, profiles and guides) with different purposes and audiences. The overview document (Part 1) [1] introduces processes, lifecycle and standardization concepts. Part 2 [10] introduces the framework and the

taxonomy. Part 3 [11] defines the process assessment guidelines and compliance requirements needed to meet the purpose of the defined VSE profiles.

The document ISO/IEC 29110-4-1 [2] provides the specification for all the Generic Profile Group profiles. The Generic Profile Group is applicable to VSEs which do not develop critical software products [1]. The Basic Profile describes the software development of a single application by a single project team with no special risk or situational factors [2]. The ISO/IEC 29110-5-1-2 document [12] provides an implementation management and engineering guide for the Basic Profile.

The ISO 29110 Set of Documents is due for approval in June 2010. It is possible that VSEs may be intimidated by this set. Moreover, this set includes ISO standards, submitted to copyright fees. However, guides are targeted at VSEs, and should be VSE-accessible, in terms of both style and cost [1].

2.3 Basic Profile

Basic Profile Processes: Objectives and Tasks Decomposition. The Basic Profile establishes VSE characteristics, needs and suggested competencies, and uses it to define process objectives. For instance, objectives related to requirements are: the SI.O2 objective "Software requirements are defined, analyzed for correctness and testability, approved by the Customer, baselined and communicated [2, p. 7]", the SI.O3 "[…] Consistency and traceability [of the design] to software requirements are established [2, p. 8]", and the SI O.4 "[…] Traceability [of the software components] to the requirements and design are established [2, p. 8]".

The Basic Profile consists of 2 processes: Project Management (PM) and Software Implementation (SI). A process is defined as "a set of interrelated or interacting activities which transforms inputs into outputs [8]". An activity is "a set of cohesive tasks of a process [8]". For each activity of the PM and SI processes, the Basic Profile details the tasks to be performed: role, description of the task, input and output products. For instance, the starting point of the 29110 use for requirement is the SI.2 "Software Requirements Analysis" activity, its list of tasks: SI.2.1 to SI.2.7 and the associated roles. Roles are: TL Technical Leader, WT Work Team, AN Analyst, and CUS Customer. Table I provides a tasks breakdown for the activity SI.2 [2, pp. 15].

Table 1. SI.2 Software requirements analysis - tasks and roles. * means (if appropriate).

Task List	Role
SI.2.1 Assign tasks to the Work Team members in accordance with their role, based on the current *Project Plan*.	TL, WT
SI.2.2 Document or update the *Requirements Specification*.	AN, CUS
SI.2.3 Verify the *Requirements Specification*.	AN
SI.2.4 Validate the *Requirements Specification*	CUS, AN
SI.2.5 Document the preliminary version of the *Software User Documentation* or update the present manual. *	AN
SI.2.6 Verify the *Software User Documentation*	AN
SI.2.7 Incorporate the *Requirements Specification,* and **Software User Documentation* to the *Software Configuration* in the baseline.	TL

Basic Profile Products. Part 29110-4-1 provides Work product specifications, and Activity input & output specification. For instance, SI.2.1 to SI.2.7 tasks have associated output products: *Requirements Specification*, *Verification Results*, *Change Request*, *Validation Results*, and [preliminary] *Software User Documentation*.

2.4 Deployment Package

Significant help is expected from Deployment Packages (DP). C. Laporte, the editor of the ISO/IEC 29110 defines a DP as "a set of artefacts developed to facilitate the implementation of a set of practices, of the selected framework, in a VSE [3]". The elements of a typical deployment package are: process description (activities, inputs, outputs, and roles), guide, template, checklist, example, presentation material, reference and mapping to standards and models, and list of tools [13]. Packages are designed in such a way that a VSE is able to implement its content without having to implement the entire framework at the same time.

Regarding requirements, the Deployment Package - Software Requirement Analysis [14] adds depth to the standard, providing guidance through a simplified breakdown of the SI.2 SW requirements analysis activity. The DP sums up the SI.2 activity in 4 tasks: requirement identification, requirements refinement and analysis, requirements verification and validation, requirements change management. For each of theses 4 tasks, the DP describes a step-by-step method.

This DP follows the SPEM approach promoted by OMG in [15]. In this DP, the tasks required for performing SW requirements analysis are defined through textual step-by-step explanations, describing how specific fine-granular development goals are achieved, through which roles, and with which resources and results. The DP also provides several templates (including a simplification of IEEE 830 [16]) of a Software Requirement Specification Document.

Training materials and an Excel-based Traceability tool can be downloaded from the public WG24 web site http://profs.logti.etsmtl.ca/claporte/English/VSE/index.html.

3 A Pilot Project on Requirements

3.1 Overview

Context of the VSE. A VSE of 14 people (with 3 software engineers) requested our help in Spring 2009. This VSE designs, builds, develops and sells a counter system intended to collect and analyze frequentation of public or private sites. Counting systems are based on stand-alone counter boxes (including sensors, power supply, data storage, and data exchange) and a software chain able to collect, analyze, present, and report counting data. The data set was downloaded from counters via infrared link or GSM, stored on PC and exchanged via a file transfer utility.

The new software project. The VSE started a complete reconstruction of its software chain in order to transform it into a web-based system called Eco-Visio, intended to host data from fleets of counting systems for each client, and able to process statistics and generate analysis reports on counting. At the end of June 2009,

the VSE hired an Information Technology graduate from our university. At the same moment in time, we initiated a pilot project intended to help the VSE implement just one part of the 29110.

The pilot project. The absence of requirement traceability and systematic testing was rapidly recognized by all stakeholders. Both authors also agreed that project management was in need of improvement, but we deliberately omitted this point. We proposed a 2-stage plan of action: - 1- implementation of the "Software Requirements Analysis" Deployment Package and - 2 - implementation of the "Software Testing" Deployment Package. The first stage is complete, and reported on in this paper.

Deployment Package. The starting point of the 29110 use for requirement is the SI.2 "Software Requirements Analysis" activity, its list of tasks - SI.2.1 to SI.2.7 - and the associated roles. A step-by-step approach to perform the required SI.2 tasks is given in the Deployment Package - Software Requirement Analysis [14]. One VSE employee received a short training course, using the training material associated with this DP, and downloaded the Traceability Tool provided with the DP. Despite all this assitance, the VSE engineer was unable to proceed with 29110 Requirements Engineering. He therefore attended a Training Session on requirements, based on the 29110 materials. A description of this session is presented in section 3.2.

Self-training packages. During the training session, the VSE engineer – like his co-trainees – attained an initial level of proficiency in using the 29110 for Requirements Specification – yet trainees asked for further assistance and guidance. We therefore constructed a dedicated assistance approach, which is presented in section 3.3. This approach relies on Self-Training Packages - a set of resources intended to develop experience skills in SE activities, e.g. Requirements Identification and SW requirement specification.

Assessment. We built 2 groups: a control group of 9 people and a study group of 10 people performing the 29110 training. We intended to measure the efficiency of the training system by comparing requirements competencies between both groups.

3.2 Training session

Training session context. We scheduled a training week on 29110 Software Requirements Analysis in December 2009. 10 young engineers (including our VSE engineer) attended the session. The 29110 Training Session comprises a course on requirements and a case study using the DP - Software Requirement Analysis [14].

Content of the training session. The session begins with an introductory lecture on requirements, but trainees are plunged into 'doing' with the preparation of a peer-review on a requirements analysis guide. This guide is issued by an ISO-9001 major software company (at which both authors had been employed for about ten years). The SW Requirements Specification (SRS) Document is issued by the DOD-STD-2167A software development standards [17]. This guide is intended to facilitate the writing of the SRS. Peer-reviewing this guide provided trainees with initial exposure to standardized requirements management.

During the second phase of the session, trainees have to contribute to the writing of a similar guide, based only on the 29110 standard. Authors provide trainees with a preliminary version of the guide, written in a top-down manner, starting from the 12207 standard processes devoted to requirements (6.4.1 Stakeholder Requirements Definition, 7.1.2 SW Requirements Analysis) to the 29110 Basic Profile SI.2 "Software Requirements Analysis" activity. Trainees have to incorporate both the DP - Software Requirement Analysis and its step-by-step approach into the guide.
Finally, trainees have to apply the enhanced guide to a 'real' SRS and update this SRS to satisfy compliance with the guide. The 'real' SRS is for eCompas - an existing system developed by the first author and former graduate students.

3.3 Towards requirements management capability

Objectives. Despite the path traced in the standard (including the guidance provided by the DP), some young engineers (and this is true of the VSE engineer in particular) may be unable to find their way through the managing requirements. Below, we present the step-by-step path proposed by the DP Requirement Analysis.
Task 1. Requirements identification. The objective is to clearly define the scope of the project and identify key requirements of the system. Steps are: (i) Collect information about the application domain; (ii) Identify project scope; (iii) Identify and capture requirements; (iv) Structure and prioritize requirements.
Task 2. Requirements refinement and analysis. The objective is to detail and analyze all the requirements identified. Steps are: (i) Detail requirements; (ii) Produce a prototype.
Task 3. Requirements verification & validation. The objective is to verify requirements and obtain validation from the customer or his representative. Steps are: (i) Clarify fuzzy requirements (verification); (ii) Review SRS (Software Requirements Specification); (iii) Validate requirements.
Task 4. Requirements change management. The objective is to manage requirements change in line with a process agreed upon with the customer. Steps are: (i) Track changes to requirements; (ii) Analyze impact of changes; (iii) Identify changes that are beyond the project scope; (iv) Prioritize changes.

The core of requirements gathering and specification must be performed in tasks 1 and 2. We decided to build two Self-Training Packages aimed at helping young engineers with: A - Requirements Identification and B - SW Requirements Specification. A discussion of Self-Training Packages is beyond the scope of this paper, but we will say that one objective of our research group is to provide VSEs with a training complement to the 29110 set of documents called the 'Self-Training Package'. Self-training packages are intended to be performed autonomously by VSE employees, requiring (almost) no interaction with a coach - except at the time of package delivery to the VSE.

The inspiration stems from the 15504-5 standard [19, Part 5] with a desire to provide software engineers with an exemplar model of software engineering activities together with complementary self-training material. While we are designing self-training for an SE activity (such as Requirements Analysis) and its required tasks

(such as Requirements identification or Requirements refinement and analysis), we aim to prescribe the engineer's tasks broken down into small units. Task definition is collected on a task card.

Fig. 1. Example of a task card.

N° 24	**Date**:	**Origin**:	**Roles assignment**	
Project :	**TASK CARD**		ANalyst	**Employee X** **Employee Y**
Process: Software Implementation (SI) **Activity**: Software Requirements Analysis (SI.2)				**Task Title**: Requirements bootstrap

WORK DESCRIPTION

Objectives

The goal of this task is to collect and identify requirements using a structured and prioritized list of requirements, and to establish a synthesis of users' needs.

Objectives are strongly related to SI.2.1 task objectives: "*The objective of this activity is to clearly define the scope of the project and identify the key requirements of the system*."

Step-by-step

1. Identify functional and technical needs
   Extract users' needs from the eCompas Statement of Work (call for tender) and the preliminary response to tender.
   Write a unique document "Needs Synthesis Document", gathering together any elements related to a functional or technical need.
2. Summarize required services (Services Identification List)
   Identify, classify and sum up users' needs through a list of high-level services required by the eCompas software.
   Each identified service (or sub-service) shall be documented with:
   - Identification number (could be temporarily left blank)
   - Type (Functional or Technical) and Domain (one of the five eCompas domain areas)
   - Service number (hierarchical numbering inside domains)
   - Actors (main users of the service)
   - Summary (a very short description of the service)
   - Origin (traceability to Statement of Work or Tender response)
   - Link to "Need Synthesis Document" (references to corresponding paragraphs)
3. Establish a glossary of the eCompas domain
4. Structure and prioritize the "Needs Synthesis Document"
   With the help of the "Services Identification List", rewrite a new version of the "Needs Synthesis Document" complying with the proposed hierarchy.
   Establish traceability.
   Number services with a hierarchical identification number.
5. Perform a peer-review of an existing SW Requirements Specification
   Prepare the review of the eCompas SRS following the instructions of the Reviewer Guide

Resources

- eCompas Statement of Work and Tendering answer
- SRS Writing Guide and Peer- Reviewer Guide

…

Output products

The main output product of this task is the "Needs Synthesis Document", which will be used in the next task - "SRS writing" as a preliminary version of the Software Requirements Specification.

Products	**V.**	**Milestone**
Needs Synthesis Document	A, B	
Services Identification List	A	

Task cards. The description of the task is designed as a theatre scene: the scene being the reference context in which the action takes place. The scene aims for unity

of place, time and action; it is a situation in which people do [and learn], a scenario of actions, a role distribution, an area mobilizing resources and means. The different components of a scene, along with their articulation, are depicted on a task card (see an example of the Requirements bootstrap card in Figure 1).
Its main elements are:

- Related 29110 Process / Activity

This reference (SI / SI.2 SW Requirements Analysis in this instance) provides a smooth link to the 29110 and through the ISP to the 12207 and 15504 standards.

- Role

Role (here ANalyst) is a quick reference to the 29110 Role

- Task Title and Objectives

Similar to Process Title, Process Purpose, and Process Outcomes as defined in ISO/IEC 12207

- Step-by-step

A comprehensive description of the work to be done - intended to be useful as a practical guide to completion of the task.

- Resources

The set of resources required. This may set up the context and/or be required to perform the task. It may include online courses that are affordable to a technology transfer centre, where the cost is beyond the reach of a VSE.

- Output products

This is generally a 29110 Work Product, or an intermediary product required to build this Work Product. A hidden goal is to initiate and develop a strategy of capitalizing on the activity, and transferring knowledge to VSE employees.

Self-training. For the self-training reported in this section, we built two task cards: Requirements bootstrap and SRS writing. Self-training is then performed as a case study: a set of resources is used to set up the context, and engineers have to perform tasks as they should do in a 'real' situation.
Our study group of 10 engineers performed both Self-Training Packages in January and February 2010. The first Training Package was intended to offer an initial level of maturity in ISO/IEC 29110 Requirements Management (through the study of SI.2 activity and a review of a ‘real’ WP11 Requirements Specification) and the second Training Package aims to perform a Requirements Analysis on a ‘real’ case. Very little interaction with the coach (the first author) occurred. Each engineer completed each package in roughly a week.

3.4 Process assessment

The Part ISO 29110-3 [11] is an Assessment Guide applicable to all VSE profiles. It is compatible with ISO/IEC 15504-2 and ISO/IEC 15504-3 [18]. As specified in [11], “a VSE-specific Process Assessment Model (PAM) can be derived by selecting only the assessment indicators in the 15504-5 Exemplar PAM, relevant to the corresponding process outcomes defined in ISO/IEC 29110-4.”

For instance, in the Basic Profile, the SI Process defines 7 objectives and SI.02 is the only one relevant to requirements: “Software requirements are defined, analyzed for correctness and testability, approved by the Customer, baselined and

communicated." [2] Then, reducing the 7.1.2 Software Requirements Analysis Process outcomes (15504 ENG.4) corresponding to the SI.02 objective will give:

1) requirements allocated to the software elements of the system and their interfaces are defined
2) software requirements are analyzed for correctness and testability
6) software requirements are approved and updated as needed
8) software requirements are baselined and communicated to all affected parties

If we apply the profile to the Base Practices of ENG.4, we can remove Base Practices that do not contribute to the selected outcomes (1, 2, 6, and 8). Hence, the list of profiled Base Practices of the ENG.4 Process is reduced to ENG.4.BP1 Specification of software requirements; ENG.4.BP3: Development of criteria for software testing; ENG.4.BP5: Evaluation and updating of software requirements; ENG.4.BP6: Communication of software requirements.

Clause 5 of ISO/IEC 15504-2 [19, Part 2] defines a measurement framework for the assessment of process capability, defined on a six point ordinal scale. 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 extent of process attribute achievement is characterized on a defined rating scale. Clause 6 of the 15504-5 [19, Part 5] presents the process capability indicators related to the process attributes associated with capability levels 1 to 5. Process capability indicators are the means of achieving the capabilities addressed by the considered process attributes.

ISO/IEC 15504 separates processes and capability levels in two dimensions whilst CMMI handles them in a single dimension. However, it should be pointed out that separate process and capability dimensions may discourage a VSE regarding process assessment. For instance, capability level 2 indicators applied to requirements relate to defining, planning, monitoring and adjusting the performance of the SI.2 Requirements Analysis activity and to identifying, defining, documenting, reviewing and adjusting each work product related to this activity. In our opinion, this kind of assessment will neither determine whether a VSE achieves the Basic Profile, nor help the VSE to improve its Requirements Engineering implementation. We would like VSE employees to understand the importance of the assessment principle, whilst performing regular self-assessment on a reduced set of major objectives. Such an objective should be formulated with a sentence in the "To be able to …" format. This proposal, applied to Requirements Engineering, is detailed in the following section.

3.5 Evaluation of the system efficiency

Since 2008, local employers in Brest have significantly increased take-up of a work placement system called "Contrat de professionnalisation" (professionalization contract) over a period of 12 months. During these 12 months, fully-paid employees attend university for approximately 250 hours of technical training (about 40 days over the whole year). This academic year, 19 young software engineers who graduated from our university in June 2009 after a 4-year programme in Computer Science or Information Technology, are benefiting from this system. As mentioned

above, 10 people chose the 29110 training; the other 9 chose to attend a UML-based analysis course. Thus, we have a population divided into 2 groups: a control group of 9 people and a study group of 10 people performing the 29110 training reported in previous sections. The UML-based analysis course and 29110 training were performed within a period of about 3 weeks between September 2009 and February 2010. Hence, we sought to measure the efficiency of the training system by comparing requirements competencies between both groups.
We defined three major objectives in requirements:

- To mobilize specification methods and tools in a real project
- To work under the control of a standardized baseline
- To produce a Software Requirement Specification (including traceability)

We decided to assess each objective on a self-assessment scale ranging from 0 to 5: - 0 - ? : Do not know anything about the topic; - 1 - Fog: has only a vague idea; - 2 - Notion: has a general idea but is unable to achieve the objective; - 3 - User: is able to achieve the objective with the help of an experienced colleague and has an initial experience of its achievement; - 4 - Autonomous: is able to work autonomously; - 5 - Expert: is able to act as an expert to modify, enrich or develop the knowledge area on which the objective focuses. We asked each of the 19 engineers to self-assess themselves three times: 1 – At job start: at the beginning of their (first) job, young engineers complete the first self-assessment; all participants did this in September 2009; 2 – At 6 months: after 6 months of employment, young engineers complete the second self-assessment; this was done in March 2010 for the whole group; 2 – At 9 months: in order to assess how software engineering practices are maturing, young engineers complete a third self-assessment in June 2010.
Table 2 presents average self-assessment scores for both groups.

Table 2. Base Practices average self-assessment scores.

	Control Group			**Study Group**		
Objectives	***Sep. 09***	***Mar. 10***	***Jun. 10***	***Sep. 09***	***Mar. 10***	***Jun. 10***
SI.2.1 To mobilize specification methods and tools in a real project	1.56	2.11	2.11	1.50	2.70	2.80
SI.2.2 To work under the control of a standardized baseline	0.78	1.44	1.44	0.70	2.60	2.60
SI.2.3 To produce a Software Requirement Specification (including traceability).	2.33	2.67	2.89	1.40	2.80	3.20

No statistical comparison was performed. Requirements training took place for both groups. However, there is evidence that self-assessment scores are increasing more significantly for the study group than for the control group.

Table 3 presents score frequency distribution for both sets.

Table 3. Base Practices self-assessment distribution.

September 2009	Control Group							Study Group						
Objectives	?	F	N	U	A	E	*Avg.*	?	F	N	U	A	E	*Avg.*
SI.2.1	2	3	1	3	0	0	*1.56*	1	4	4	1	0	0	*1.5*
SI.2.2	3	5	1	0	0	0	*0.78*	5	3	2	0	0	0	*0.7*
SI.2.3	0	1	4	4	0	0	*2.33*	2	4	2	2	0	0	*1.4*
March 2010	Control Group							Study Group						
Base Practice	?	F	N	U	A	E	*Avg.*	?	F	N	U	A	E	*Avg.*
Objectives	1	1	4	2	1	0	*2.11*	0	0	4	5	1	0	*2.7*
SI.2.2	2	2	4	2	1	0	*1.44*	0	0	5	4	1	0	*2.6*
SI.2.3	0	1	2	5	1	0	*2.67*	0	0	3	6	1	0	*2.8*
June 2010	Control Group							Study Group						
Objectives	?	F	N	U	A	E	*Avg.*	?	F	N	U	A	E	*Avg.*
SI.2.1	1	1	4	2	1	0	*2.11*	0	0	3	6	1	0	*2.8*
SI.2.2	2	2	4	1	0	0	*1.44*	0	0	5	4	1	0	*2.6*
SI.2.3	0	0	3	4	2	0	*2.89*	0	0	3	5	2	0	*3.2*

Empirical evaluation. The VSE engineer reported that he was now ready to apply the SI.2 SW Requirements Analysis on the Eco-Visio project. As the specifications were soon established by another VSE colleague, he only reviewed and rewrote some sections of the existing Requirements Specification, in order to establish compliance with the template provided in the DP - Software Requirement Analysis [14]. Once updated, the WP11 Requirement Specification [Validated] served as an input to the SI.5 SW Integration and Tests. The system has been deployed since April 2010 and load testing and application optimization should be soon complete. Defects have to be corrected through a short cycle of SI activities.
As an empirical measure of its satisfaction, the VSE asked for a similar approach for the SI.5 SW Integration and Tests. In particular, the VSE wants guidance and support in establishing a disciplined Change Request Process. A Self-Training Package is under construction, and we should start with the "Software Testing" DP [19] as a basis for the whole Training Package. Probably because Tests occur in many SE activities, this DP is organized so that it spans PM and SI tasks, raising a wealth of new questions.

5 Conclusion and future work

We reported on a system that was intended to help a VSE with requirements management. Two points are discussed (1) a Training Session based on 29110 materials; (2) Self-Training Packages intended to perform requirements definition and analysis through a step-by-step approach. We used self-assessment to establish a comparison between a control group of 9 people attending a UML-based analysis course and our 10-person study group performing our proposition. Self-assessment scores are increasing more significantly for the study group than for the reference set. The concept of the Self-Training Package seems to extend to other processes such as design or testing. Further work is required to determine how far the scope of this concept and its main tool - task cards - can be extended.

References

1. International Organization for Standardization (ISO): ISO/IEC DTR 29110-1 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) -- Part 1: Overview. ISO, Geneva (2010)
2. ISO: ISO/IEC FDIS 29110-4-1 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) -- Part 4: Specification-VSE Generic Profile Group. ISO, Geneva (2010)
3. Laporte, C. Y.: Contributions to Software Engineering and the Development and Deployment of International Software Engineering Standards for Very Small Entities. PhD thesis of the Université de Bretagne Occidentale, Brest (2009), http://tel.archives-ouvertes.fr/tel-00483255/fr/
4. Richardson, I. , Wangenheim, C. G.: Why are Small Software Organizations Different?. IEEE Software 24 (1), pp.18-22 (2007)
5. Calvo-Manzano J. A. et al.: Experiences in the Application of Software Process Improvement in SMES. Software Quality Journal 10 (3), pp. 261-273 (2002)
6. Hardiman, S.: REDEST - 14 Best Practice SME Experiments with Innovative Requirements Gathering Techniques. In: Proceedings of the IEEE Joint International Conference on Requirements Engineering (RE'02), pp. 191 (2002)
7. CARSA: Redest Requirements Engineering Best Practice Case Book - Experiment-based cross-fertilisation and dissemination of software requirements gathering techniques. http://www.redest.net/documentos/LibretoRedest.zip (2003)
8. ISO: ISO/IEC 12207:2008 Information technology -- Software life cycle processes. ISO, Geneva (2008)
9. ISO: ISO/IEC 15289:2006, "Systems and software engineering -- Content of systems and software life cycle process information products (Documentation)". ISO, Geneva (2008)
10. ISO: ISO/IEC FDIS 29110-2 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) -- Part 2: Framework and Taxonomy. ISO, Geneva (2010)
11. ISO: ISO/IEC DTR 29110-3 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) -- Part 3: Assessment Guide. ISO, Geneva (2010)
12. ISO: ISO/IEC DTR 29110-5-1-2 Software Engineering - Lifecycle Profiles for Very Small Entities (VSEs) -- Part 5: Management and Engineering Guide-Basic VSE Profile. ISO, Geneva (2010)
13. Laporte, C.Y., Alexandre, S., O'Connor, R.: A Software Engineering Lifecycle Standard for Very Small Enterprises. In: R.V. O'Connor et al (eds) EuroSPI 2008. CCIS, vol. 16, pp. 129-141. Springer-Verlag, Heidelberg (2008)
14. Alexandre, S., Laporte, C. Y.: Deployment Package - Software Requirement Analysis. Available from http://profs.logti.etsmtl.ca/claporte/VSE/Publications/DP-Software Requirements Analysis-V1_2.doc (2010)
15. OMG, "Software & Systems Process Engineering Meta-Model Specification Version 2.0", http://www.omg.org/cgi-bin/doc?formal/08-04-01.pdf (2004)
16. IEEE: IEEE Std 830:1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE (1998)
17. Department of Defence: Defense System Software Development, DOD-STD-2167A. 29 February 1998 (1998)
18. ISO: ISO/IEC 15504 Information technology -- Process assessment. ISO, Geneva (2004)
19. Gómez Arenas, L.: Deployment Package - Software Testing. Available from http://profs.logti.etsmtl.ca/claporte/VSE/Publications/DP-Software_Basic_Profile_Testing-CL00.doc (2010)

Modelling by Example:
Requirements engineering during the bidding stage of dialog-oriented software projects

Axel Kalenborn

Universität Trier, Wirtschaftsinformatik, Campus II,
54286 Trier, Germany
axel.kalenborn@uni-trier.de

Abstract: Before a software project is implemented, there is a stage that has until now received little consideration in literature: the bidding stage, during which a rough concept of the software to be implemented has to be established in the form of a bid. Part of a bid is a cost estimate that should be as precise as possible. During the bidding stage, bidders are not being paid and compete with each other, which means they have to work under great pressure of time, success and cost. The costs caused by the common methods of requirements engineering are too high, so that these methods cannot be used at this stage yet. The following paper pictures an approach to facilitate the bidding stage of dialog-oriented applications and illustrates it by means of a tool.

Keywords: dialog-oriented software projects, bidding stage, mock-up, enriched mock-ups, software calculation

1. Introduction

In the common process models of software engineering, after signing of the contract, projects generally begin with a detailed stage of requirements engineering and then pass on to conception and realization [3], [12]. During these stages, methods of requirements engineering are available by means of which the requirements regarding the software to be implemented can be specified in collaboration with the principal [9].

However, the initial requirements analysis and the documentation thereof actually do not take place only after signing of the contract but already before the start of the software project within the framework of bid preparation. It is primarily used to enhance the comprehension of the project and provides a basis for the bid and/or the contract to be concluded with a prospective customer.

During the bidding stage, detailed requirements analyses are not yet possible because the analysis is not being paid for and the bidders are competing with other suppliers. If a bidder is not awarded the contract for the software project, the incurred expenses are not covered. The possible amount of this investment depends on the bidder's estimation regarding the acceptance of his bid and is hard to determine. IT budgets are running short and require an efficient approach when preparing a bid. The present paper addresses this problem and presents a way how to facilitate the bidding

stage in the dialog-oriented application area, especially in web-projects, like portals, company web pages or internet based applications.

2. Preparation of software projects

The process of initiating a business relationship for a specific project is supposed to be described as the preparation of software projects. This process is commonly called sales phase. From the point of view of the principal, the objective of this process is to assign a software project to the best implementation partner. Usually, this process is characterized by one buyer and several bidders applying for the job. Each of the potential agents wants to get the project and be awarded the contract; they are competing with each other.

The preparation process involves risks and uncertainties both for the principal and the agent. [12]. There is an asymmetry of information between the two parties. The potential agent doesn't know the requirements and the budget, the principal only has a limited notice of the agent's actual reputation in the project field. Therefore, both partners try to exchange the information required for decision-making. For the agent, this means that he has to prepare a bid and demonstrate his capacity.

2.1 Central problems when preparing a bid

A bid contains the scope and conditions of a software project and thus defines the legal framework for its implementation. The following paragraphs are supposed to outline the central problems during bid preparation.

Pressure of time and cost during bid preparation

When preparing a bid, companies find themselves in a difficult situation. On the one hand, the bids provide a basis for a successful project implementation and therefore must be as precise, complete and correct as possible. On the other hand, these very characteristics require great effort. The possible effort that can be invested for the participation in an awarding to be worthwhile depends on the scope of the order and on the chance to be awarded the contract. The chance to be actually awarded the contract for a project on the other hand depends on different factors, such as competence, creativity etc. [2]. An essential point, however, is the number of participating bidders. If there are 10 bidders, a success rate of approximately 10 % can be assumed, because your bid has to be better than 9 others.

This causes a great pressure during the preparation of the bid. Often the bids cannot be prepared as carefully as is necessary but are rough estimates with high risks. Small and medium-sized companies are at a disadvantage in comparison to large-scale enterprises because the bids are mostly being prepared by the company management itself and the margins of the projects are lower [7]. So the time and money they are able to spend in a bid is very limited.

Professionalization in the competitive environment

If you want to have success in selling a project, the customer not only has to understand what you want to build and how you want to do it, he has to feel confident that you are the right partner. To transport this, a visualization of what you want to do is helpful and often necessary, especially in the area of web applications, where the layout and the look and feel have great influence on the decision makers.

Therefore, in practice, the presentation view is an important basis for the dialog with the customer [5]. This is due to the fact that the decision makers are often not able or not willing to understand abstract models or descriptions.

In many cases, the visualization is also the only element of a project that the decision makers can discuss because the technical details of an application are so complex that only IT professionals can understand and assess them. This phenomenon is known as IKIWISI and deals with the problem that the software users do not understand the requirements until they see them [1].

Because of this, drafts, screenshots and HTML prototypes are standard in the context of bid preparation of internet based projects today and have to be created to keep the chance.

Lacking IT affinity of the decision makers

Information technology has an increasing influence on business processes and the business success of companies. Therefore, IT-related decisions must be taken by those who are involved in these business processes. Consequently, the specialist departments of the companies have an increasing influence on IT-related decisions. This is mainly due to the fact that information and communication technologies are nowadays more closely connected to the business processes of the companies and are thus of an increasing importance for smooth business processes [4].

Bidding document problems

A bid is supposed to finally convince the customer. The second place in a bid invitation or a competitive presentation is just as dissatisfying as the last place. So the bidding documents have to convince the customer of the bidder's competence and of the idea that he is the right partner for the respective project. This can only be reached with professional and elaborate bids, the implementation of which, however, causes great effort.

2.2 Requirements regarding a method to facilitate bid preparation

The methods of requirements engineering are applicable for the preparation of software projects in the context of the bidding stage only to a limited extent. Many common methods are too complex and their results are too abstract and thus not suitable to convince decision makers.

This problem has already been recognized in practice. So an important factor today is the visualization of applications. Particularly when it comes to the implementation of web-based projects, the design is at least just as important as the

functional description of the application. Customers wish to see a presentation of the web pages to be created already at the stage of bid preparation. These so-called "mock-ups" provide a limited preview of the application to be created, are used in order to explain functions and to design the application to be created and thus build trust in the implementation capabilities of the bidder.

The creation of the mock-ups is often not a very efficient process in practice. The mock-ups are designed in image processing applications, extended and/or explained in presentation programs, completed by the required specifications in a word processing program and calculated as a bid in a spreadsheet program.

Therefore, a new approach to the requirements analysis and documentation is needed. The question is how to set up mock-ups in order to obtain more than visualization. A possible solution is the semantic enrichment of the mock-ups in order to use them both for visualization and bidding or specifications preparation.

3. The idea of Modelling by Example

The so-called „by example" approaches derive from the database context and were introduced as "Query by Example (QbE)" [13], [8]. The QbE approach is an alternative to the Structured Query Language (SQL) and is used in a modified form in many applications. Databases such as Microsoft Access or Corel Paradox for example facilitate the switching between a QbE view and an equivalent SQL view of queries. The QbE technique enables even non-professionals who are unfamiliar with database languages to carry out complex evaluations or changes in relational databases.

Modelling by Example (MbE) is supposed to transfer this idea to the modeling of web sites and web applications. Instead of asking the user to learn new methods or display formats, the result, i.e. the view of the user interface, is used as the starting point for the conception of the application [6]. Since additional information can be included, the model can later be refined and completed by experienced professionals.

3.1 Objective of Modelling by Example

The objective of MbE is to support the staff members entrusted with the preparation of the bid during the stage of preparation of dialog-oriented software projects. During this stage, the requirements regarding the software to be implemented have to be collected precisely, so that a competitive, convincing bid can be created.

The principal provides information and must therefore be involved in the requirements engineering. But he often faces restrictive timely limits, because he has to deal with several bidders that need information in equal measure. Thus, the basis for the requirements engineering is provided by a more or less precise specification sheet, a briefing document or just the notes taken during a meeting. So, despite this lack of information and the great pressure of time, a convincing description of the project to be implemented must be created for the principal.

Decisions regarding the awarding of projects are mostly taken in committees, the so-called buying centers. The participants of the buying centers have different points of view regarding the project, for example costs, look and feel or technical implementation. In order to be able to convince as many members of the buying

center as possible, the bid is often extended by means of additional documents, e.g. a technical specification or a graphic concept.

This means that one document is not sufficient because it cannot contain all points of view in equal measure. The basis for the signing of the contract is the bidding document containing the way the agent intends to implement the project as well as the price for this implementation. The bid is usually presented to the customer, so another document in the form of a presentation is needed.

Today, all these documents are manually created in graphical software, word processing programs, spreadsheet programs and presentation programs, causing great effort. This is where the MbE idea comes in, because it is supposed to facilitate significant potential savings by means of efficient mock-ups.

3.2 Realization of Modelling by Example

MbE is being used to be able to efficiently create and semantically enrich mock-ups so that they can be used for different purposes. The creation of the mock-ups is supposed to include regular consultation with the customer if possible in order to get a direct feedback regarding the implementation and avoid elaborate minutes and complex descriptions [1].

The design of the mock-ups corresponds to the widely-used template concept of web applications. It is based on a uniform layout, the so-called template or master which describes the page structure and the graphic design the contents and applications are going to be integrated into. A template contains the navigation structure of the website and defines the look and feel that is going to be found on all subpages.

The contents of a web site consist of text and multimedia-based elements such as images or animated graphics which are structured by means of formatting. In the prototype version, placeholders for the picture elements and captions for the structuring of the pages and the exemplary representation are used [11].

Apart from textual and other contents, web sites consist of interactive modules, e.g. dialogs. Standard elements of interactive applications are dynamic tables, input fields, buttons, list boxes, radio buttons etc. One example for this is a contact form. It contains input fields, a button to transfer data and possibly a CAPTCHA element in order to avoid SPAM. Additionally, rules for syntax checking of the input fields and reactions to faulty entries can be defined. A contact form can be found on almost any website and is thus a good example for a re-usable module.

Re-usable modules are a key component in the MbE approach. In order to be able to define a mock-up as quickly as possible, a set of templates for modules is important, so that components such as a picture gallery, a site map or interactive location plans can be easily integrated into the prototype. The MbE tool contains a collection of modules with frequently used components that can be individually extended.

The modules are designed in a way that allows for them to be integrated into the templates, so that they adjust themselves to the look and feel of the website. The more re-usable components a project contains, the faster a mock-up can be created. However, the re-usable modules are not limited to specific applications but also

contain prepared input masks with a certain number of fields or multistage wizards which can easily be adjusted in dialog with the customer.

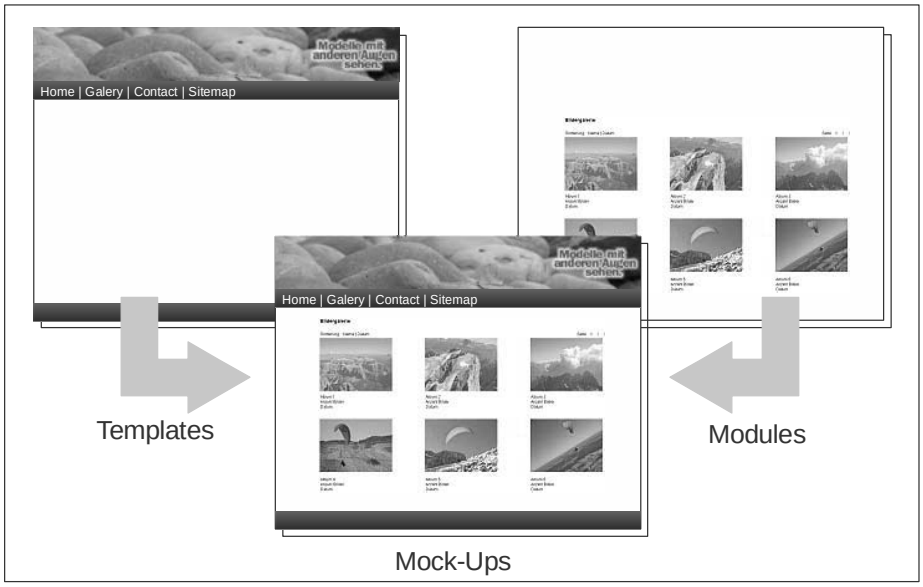


Figure 1: Modules and templates.

An essential factor during the creation of mock-ups is the precision of the graphic implementation. This is something that the tool attaches great importance to, because the output is supposed to comply exactly with the graphic designers' specifications, so that criticism on avoidable display errors can be prevented from the start. Another important point is the separation of conceptual and graphic design. In the MbE approach, the look and feel of the website is defined by means of the templates that can be stored for each page or module. Thus, the integration of modules and their technical description can be separated from the design, so that unneeded loops and further inquiries during the bidding process can be avoided. The modeled mock-ups are automatically clickable, so that customers can use them in an interactive way. Furthermore, they can be transformed into presentations or videos the customer can watch in order to better understand his web application.

3.3 Results of Modelling by Example

In order to be able to create a bid from the mock-up, for one thing, the technical description of the modules, pages and templates and for another thing, the calculation of the costs for the implementation is needed. Another aspect is the time scheduling for the realization of the project.

In order to store these characteristics, specifications can be assigned to each element of a mock-up. Specifications are functional or technical descriptions for the bid and the subsequent implementation as they are common in specification sheets.

The specifications are assigned in the form of texts and are meant to explain the functions of the mock-ups. Also in this case, the objective is to be able to write a bid as quickly as possible and fall back on predefined specifications. Thus, the technical description of a picture gallery has to be defined in the template only once and can then be reused in all the bids.

The calculation is based on a pattern similar to that of the technical description. In the calculation of software projects, different types of costs have to be considered, e.g. the license fees for a database, the required man-days for implementing an application or testing a component [10]. These types of costs and their respective cost rates are set in the tool and then used for the calculation of the modules. The schema which is implemented here connects the types of costs with a time dimension and a cost rate.

For the calculation within the scope of the mock-ups or modules, the predefined cost rates are furnished with values. If for example for the integration of a picture gallery into the project, the purchasing of a license is required and 3 man-days are needed for the graphic and technical adjustment, these costs will be stored in the module and calculated with the rates defined in the project. The cost estimate within the scope of the predefined modules is thus at first carried out on an abstract level which is then furnished with the defined cost rates in the respective project and calculated. The modeler always has an updated status of the costs of his software project and can individually adjust them in the mock-up. This is supposed to significantly increase the transparency and precision of the cost estimate.

The specific output of the modeling is a configurable report which provides the basis for the bid to be created. This report combines the pages of the mock-up with the stored technical descriptions and the calculations to form a bidding framework which can then be revised in Word or Open Office. Further reports are imaginable, for example in the form of performance or requirements specifications which also integrate the stored technical specifications of the mock-up and the modules.

4. The Modelling by Example tool

The MbE tool is implemented as a platform independent Java application and demonstrates the applicability of the concept in practice.

The MbE tool can be characterized as a combination of graphic program and requirements engineering tool. In the central area, the user interfaces are designed. For this purpose, mask elements such as text boxes and buttons or forms such as rectangles, lines and polygons are available; bitmaps can be imported or placed by copy and paste. The site structure is designed as a tree. A mock-up contains different modules which in turn are composed of several pages. All the pages can be linked to each other and it is possible to display functions.

The specifications are defined via the properties of the objects. This is where the functional and technical descriptions as well as the estimated costs for the implementation are stored in a structured way.

When generating reports, such as bids or functional descriptions, the tool combines the mock-up screens with their descriptions and calculations and groups them by the defined modules. The generated report is an RTF document which can be seen as basis for the bid or the functional description. It can be completed or revised in text processing software.

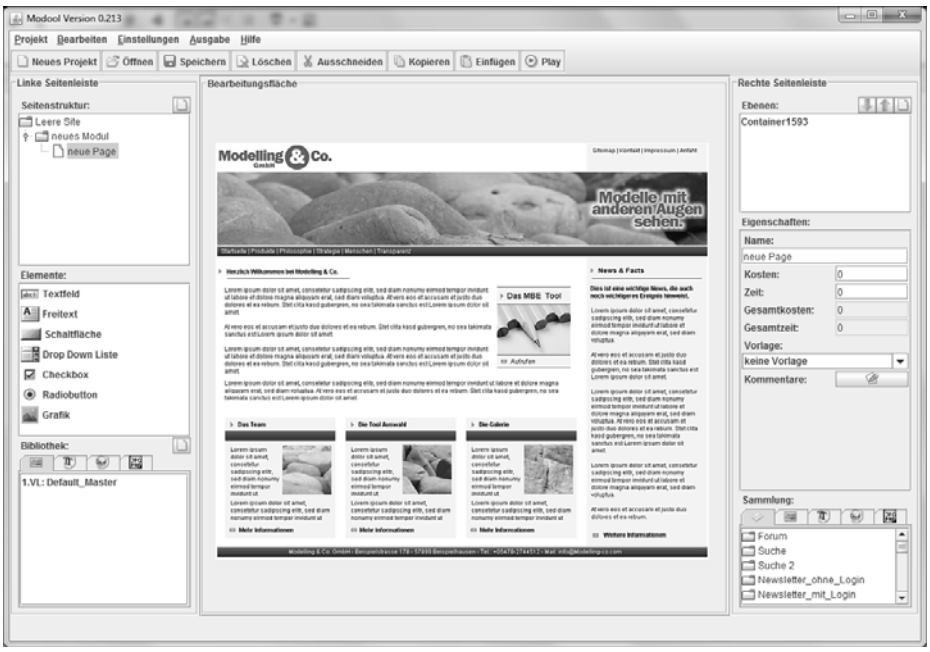


Figure 2: The MbE modeling tool.

The reusable modules are combined to a collection. All elements of the collection can directly taken over into the current project using a drag-and-drop mechanism. The tool implements a multistage template concept in which an arbitrary number of templates can be nested, so that even complex layer structures can be realized.

5. Summary and prospects

The MbE approach focuses on the preparation of dialog-oriented software projects, especially in the web context. In this context, a professional bid is to be created which has to be furnished with accompanying documents and presentations.

The MbE approach consistently follows the requirements found in the field and focuses on the creation of a mock-up for the preparation of a bid because visualization of an application is often the only way to be able to discuss IT systems with an expert representative. In the process, the idea of single source publishing is taken up in order to be able to use the created mock-ups for the bid preparation, the presentation or the interactive use by the customer.

The MbE tool is an auxiliary means for the early requirements analysis and bid preparation and is primarily used to increase efficiency in the bidding process. It is meant to help create cost estimates as quickly as possible which can then be verified by means of other methods of cost estimation [10].

The limitations of the approach are beyond the domain of dialog-oriented systems e.g. in the area of algorithmic applications or embedded systems. The focus is on websites and web applications that ideally contain standardized modules. The more of

these standardized modules a project contains and the more of these modules can be reused, the feaster we are able to build mock-ups, their related reports and calculations. But approaches have to be found in other areas in order to design the preparation of the projects more efficiently.

An evaluation of the method and the tool is intended to be carried out in collaboration with practice partners.

List of references

1. Boehm, B.: Requirements that Handle IKIWISI, COTS, and Rapid Change, IEEE Computer, 33 (7): Seite 99-102, IEEE Computer Society, New York (2000)
2. Brennan, R./Canning, L./McDowell, R.: Business-to-Business-Marketing, SAGE Publications Ltd, London (2008)
3. Dumke, R.: Modernes Software Engineering, eine Einführung, Vieweg-Verlag, Magdeburg (1993)
4. Immes, S.: Wahrgenommenes Risiko bei der industriellen Kaufentscheidung, Trier (1993)
5. Lauesen, S: User interface design: a software engineering perspective, Addison Wesley, Copenhagen (2005)
6. Lechner, S.: Web-Scheme Transformers By-Example, Johann Kepler Universität, Linz (2004)
7. Meffert, H.: Marketing: Grundlagen marktorientierter Unternehmensführung, 9. Auflage, Gabler-Verlag, Münster (2000)
8. Myers, B. A.: Visual Programming, programming by example, and program visualization: a taxonomy, Proceedings of the SIGCHI conference on Human factors in computing systems, Seite 59-66, ACM, New York (1986)
9. Pohl, K.: Requirements Engineering: Grundlagen, Prinzipien, Techniken, 2. Auflage, Dpunkt-Verlag, Essen (2008)
10. Sneed, H. M.: Software Projektkalkulation, praxiserprobte Methoden der Aufwandsschätzung für verschiedene Projektarten, Hanser-Verlag, Budapest (2005)
11. Thaller, G. E.: Software-Anforderungen für Web Projekte, Galileo Computing Nürnberg (2002)
12. Weiber, R./Jacob, F.: Kundenbezogene Informationsgewinnung, in: Kleinaltenkamp, M./Plinke, W. (Hrsg.): Technischer Vertrieb - Grundlagen, 2. Aufl., Springer-Verlag, Berlin, S. 523-612 (2000)
13. Zloof, M. M.: Query-by-Example: a data base language, IBM Systems Journal, 16 (4): Seite 324-343, IBM Corporation, New York (1977)

Previously published ICB - Research Reports

2010

No 39 (May 2010)
Schauer, Stefan; Heise, David, Frank, Ulrich: "Entwurf einer Mentoring-Konzeption für den Studiengang M.Sc. Wirtschaftsinformatik an der Fakultät für Wirtschaftswissenschaften der Universität Duisburg-Essen"

No 38 (February 2010)
Schauer, Carola: "Wie praxisorientiert ist die Wirtschaftsinformatik? Einschätzungen von CIOs und WI-Professoren"

No 37 (January 2010)
Benavides, David; Batory, Don; Grunbacher, Paul (Eds.): "Fourth International Workshop on Variability Modelling of Software-intensive Systems"

2009

No 36 (December 2009)
Strecker, Stefan: "Ein Kommentar zur Diskussion um Begriff und Verständnis der IT-Governance - Anregungen zu einer kritischen Reflexion"

No 35 (August 2009)
Rüngeler, Irene; Tüxen, Michael; Rathgeb, Erwin P.:"Considerations on Handling Link Errors in STCP"

No 34 (June 2009)
Karastoyanova, Dimka; Kazhamiakan, Raman; Metzger, Andreas; Pistore, Marco (Eds.): "Workshop on Service Monitoring, Adaption and Beyond"

No 33 (May 2009)
Adelsberger,Heimo; Drechsler , Andreas; Bruckmann, Tobias; Kalvelage, Peter; Kinne, Sophia; Pellinger, Jan; Rosenberger, Marcel; Trepper, Tobias: „Einsatz von Social Software in Unternehmen – Studie über Umfang und Zweck der Nutzung"

No 32 (April 2009)
Barth, Manfred; Gadatsch, Andreas; Kütz, Martin; Rüding, Otto; Schauer, Hanno; Strecker, Stefan: „Leitbild IT-Controller/-in – Beitrag der Fachgruppe IT-Controlling der Gesellschaft für Informatik e. V."

No 31 (April 2009)
Frank, Ulrich; Strecker, Stefan: "Beyond ERP Systems: An Outline of Self-Referential Enterprise Systems – Requirements, Conceptual Foundation and Design Options"

No 30 (February 2009)
Schauer, Hanno; Wolff, Frank: „Kriterien guter Wissensarbeit – Ein Vorschlag aus dem Blickwinkel der Wissenschaftstheorie (Langfassung)"

No 29 (January 2009)
Benavides, David; Metzger, Andreas; Eisenecker, Ulrich (Eds.): "Third International Workshop on Variability Modelling of Software-intensive Systems"

2008

No 28 (December 2008)
Goedicke, Michael; Striewe, Michael; Balz, Moritz: „Computer Aided Assessments and Programming Exercises with JACK“

No 27 (December 2008)
Schauer, Carola: "Größe und Ausrichtung der Disziplin Wirtschaftsinformatik an Universitäten im deutschsprachigen Raum - Aktueller Status und Entwicklung seit 1992"

No 26 (September 2008)
Milen, Tilev; Bruno Müller-Clostermann: " CapSys: A Tool for Macroscopic Capacity Planning"

No 25 (August 2008)
Eicker, Stefan; Spies, Thorsten; Tschersich, Markus: "Einsatz von Multi-Touch beim Softwaredesign am Beispiel der CRC Card-Methode"

No 24 (August 2008)
Frank, Ulrich: "The MEMO Meta Modelling Language (MML) and Language Architecture – Revised Version"

No 23 (January 2008)
Sprenger, Jonas; Jung, Jürgen: "Enterprise Modelling in the Context of Manufacturing – Outline of an Approach Supporting Production Planning"

No 22 (January 2008)
Heymans, Patrick; Kang, Kyo-Chul; Metzger, Andreas, Pohl, Klaus (Eds.): "Second International Workshop on Variability Modelling of Software-intensive Systems"

2007

No 21 (September 2007)
Eicker, Stefan; Annett Nagel; Peter M. Schuler: "Flexibilität im Geschäftsprozessmanagement-Kreislauf"

No 20 (August 2007)
Blau, Holger; Eicker, Stefan; Spies, Thorsten: "Reifegradüberwachung von Software"

No 19 (June 2007)
Schauer, Carola: "Relevance and Success of IS Teaching and Research: An Analysis of the ‚Relevance Debate'

No 18 (May 2007)
Schauer, Carola: "Rekonstruktion der historischen Entwicklung der Wirtschaftsinformatik: Schritte der Institutionalisierung, Diskussion zum Status, Rahmenempfehlungen für die Lehre"

No 17 (May 2007)
Schauer, Carola; Schmeing, Tobias: "Development of IS Teaching in North-America: An Analysis of Model Curricula"

No 16 (May 2007)
Müller-Clostermann, Bruno; Tilev, Milen: "Using G/G/m-Models for Multi-Server and Mainframe Capacity Planning"

No 15 (April 2007)
Heise, David; Schauer, Carola; Strecker, Stefan: "Informationsquellen für IT-Professionals – Analyse und Bewertung der Fachpresse aus Sicht der Wirtschaftsinformatik"

No 14 (March 2007)
Eicker, Stefan; Hegmanns, Christian; Malich, Stefan: "Auswahl von Bewertungsmethoden für Softwarearchitekturen"

No 13 (February 2007)
Eicker, Stefan; Spies, Thorsten; Kahl, Christian: "Softwarevisualisierung im Kontext serviceorientierter Architekturen"

No 12 (February 2007)
Brenner, Freimut: "Cumulative Measures of Absorbing Joint Markov Chains and an Application to Markovian Process Algebras"

No 11 (February 2007)
Kirchner, Lutz: "Entwurf einer Modellierungssprache zur Unterstützung der Aufgaben des IT-Managements – Grundlagen, Anforderungen und Metamodell"

No 10 (February 2007)
Schauer, Carola; Strecker, Stefan: "Vergleichende Literaturstudie aktueller einführender Lehrbücher der Wirtschaftsinformatik: Bezugsrahmen und Auswertung"

No 9 (February 2007)
Strecker, Stefan; Kuckertz, Andreas; Pawlowski, Jan M.: "Überlegungen zur Qualifizierung des wissenschaftlichen Nachwuchses: Ein Diskussionsbeitrag zur (kumulativen) Habilitation"

No 8 (February 2007)
Frank, Ulrich; Strecker, Stefan; Koch, Stefan: "Open Model - Ein Vorschlag für ein Forschungsprogramm der Wirtschaftsinformatik (Langfassung)"

2006

No 7 (December 2006)
Frank, Ulrich: "Towards a Pluralistic Conception of Research Methods in Information Systems Research"

No 6 (April 2006)
Frank, Ulrich: "Evaluation von Forschung und Lehre an Universitäten – Ein Diskussionsbeitrag"

No 5 (April 2006)
Jung, Jürgen: "Supply Chains in the Context of Resource Modelling"

No 4 (February 2006)
Lange, Carola: "Development and status of the Information Systems / Wirtschaftsinformatik discipline: An interpretive evaluation of interviews with renowned researchers, Part III – Results Wirtschaftsinformatik Discipline"

2005

No 3 (December 2005)

Lange, Carola: "Development and status of the Information Systems / Wirtschaftsinformatik discipline: An interpretive evaluation of interviews with renowned researchers, Part II – Results Information Systems Discipline"

No 2 (December 2005)

Lange, Carola: "Development and status of the Information Systems / Wirtschaftsinformatik discipline: An interpretive evaluation of interviews with renowned researchers, Part I – Research Objectives and Method"

No 1 (August 2005)

Lange, Carola: „Ein Bezugsrahmen zur Beschreibung von Forschungsgegenständen und -methoden in Wirtschaftsinformatik und Information Systems"

Research Group	Core Research Topics
Prof. Dr. H. H. Adelsberger Information Systems for Production and Operations Management	E-Learning, Knowledge Management, Skill-Management, Simulation, Artificial Intelligence
Prof. Dr. P. Chamoni MIS and Management Science / Operations Research	Information Systems and Operations Research, Business Intelligence, Data Warehousing
Prof. Dr. F.-D. Dorloff Procurement, Logistics and Information Management	E-Business, E-Procurement, E-Government
Prof. Dr. K. Echtle Dependability of Computing Systems	Dependability of Computing Systems
Prof. Dr. S. Eicker Information Systems and Software Engineering	Process Models, Software-Architectures
Prof. Dr. U. Frank Information Systems and Enterprise Modelling	Enterprise Modelling, Enterprise Application Integration, IT Management, Knowledge Management
Prof. Dr. M. Goedicke Specification of Software Systems	Distributed Systems, Software Components, CSCW
Prof. Dr. T. Kollmann E-Business and E-Entrepreneurship	E-Business and Information Management,E-Entrepreneurship/ E-Venture, Virtual Marketplaces and Mobile Commerce, Online-Marketing
Prof. Dr. B. Müller-Clostermann Systems Modelling	Performance Evaluation of Computer and Communication Systems, Modelling and Simulation
Prof. Dr. K. Pohl Software Systems Engineering	Requirements Engineering, Software Quality Assurance, Software-Architectures, Evaluation of COTS/Open Source-Components
Prof. Dr.-Ing. E. Rathgeb Computer Networking Technology	Computer Networking Technology
Prof. Dr. A. Schmidt Pervasive Computing	Pervasive Computing, Uniquitous Computing, Automotive User Interfaces, Novel Interaction Technologies, Context-Aware Computing
Prof. Dr. R. Unland Data Management Systems and Knowledge Representation	Data Management, Artificial Intelligence, Software Engineering, Internet Based Teaching
Prof. Dr. S. Zelewski Institute of Production and Industrial Information Mangement	Industrial Business Processes, Innovation Management, Information Management, Economic Analyses

For more information visit us on the Web: **http://www.icb.uni-due.de**

ISSN 1860-2770 (Print)
ISSN 1866-5101 (Online)