ICB
Institut für Informatik und Wirtschaftsinformatik

Brian Berenbach, Maya Daneva,
Jörg Dörr, Samuel Fricker,
Vincenzo Gervasi, Martin Glinz,
Andrea Herrmann, Benedikt Krams,
Nazim H. Madhavji, Barbara Paech,
Sixten Schockert, Norbert Seyff (Eds.)

17th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2011)

ICB-RESEARCH REPORT

Proceedings of the REFSQ 2011 Workshops REEW, EPICAL and RePriCo, the REFSQ 2011 Empirical Track (Empirical Live Experiment and Empirical Research Fair), and the REFSQ 2011 Doctoral Symposium

UNIVERSITÄT DUISBURG ESSEN

ICB-Research Report No. 44
September 2011

Proceedings Edited By:

Brian Berenbach
Maya Daneva
Jörg Dörr
Samuel Fricker
Vincenzo Gervasi
Martin Glinz
Andrea Herrmann
Benedikt Krams
Nazim H. Madhavji
Barbara Paech
Sixten Schockert
Norbert Seyff

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 events which were held during the Requirements Engineering: Foundation for Software Quality (REFSQ) conference 2011 in Essen, Germany.

- Requirements Engineering Efficiency Workshop (REEW)
- Requirements Prioritization for customer-oriented Software-Development (RePriCo)
- Workshop on Empirical Research in Requirements Engineering: Challenges and Solutions (EPICAL)
- Empirical Research Fair
- Empirical Live Experiment
- Doctoral Symposium

Table of Contents

Part I

REFSQ 2011 Workshop Proceedings

1 Preface

REFSQ 2011 has continued the newly established tradition of organizing some workshops in conjunction with the conference. This year, we have held three half-day workshops.

The Requirements Engineering Efficiency Workshop (REEW 2011) was organized by Samuel Fricker and Norbert Seyff. The Program Committee for this workshop has accepted six papers for presentation, addressing the topics of efficiency tactics and efficiency practice in Requirements Engineering.

RePriCo'11, the Workshop on Requirements Prioritization for customer-oriented Software-Development, organized by Benedikt Krams and Sixten Schockert, aimed at providing a platform for presenting and discussing new and innovative approaches to requirements prioritization. Five papers were presented.

EPICAL 2011, the Workshop on Empirical Research in Requirements Engineering: Challenges and Solutions was organized by Andrea Herrmann and Maya Daneva. The main goal of this workshop was to create a forum for debating needs, benefits, and challenges in using empirical approaches in RE. The workshop program featured a keynote talk by Kurt Schneider, a contributed paper, and an expert panel.

Initially, we had planned a fourth workshop on Requirements Engineering in small companies. Unfortunately, this workshop had to be canceled due to an insufficient number of submissions.

With a total of 42 registered participants, the REFSQ 2011 workshops were highly successful. I hope that these workshop proceedings will help to further disseminate the ideas, concepts, and results presented in these workshops.

I thank the organizers and the program committees of the three workshops for their effort to set up these workshops, solicit submissions and selecting submissions for inclusion in the workshop program. I also thank Vanessa Stricker for setting up and maintaining the website and Philipp Schmidt for producing this proceedings volume.

Zurich, 20 June 2011

Martin Glinz
REFSQ 2011 Workshops Chair

2 Workshop Calls

Requirements Engineering Efficiency Workshop

1st International
Requirements Engineering
Efficiency Workshop (REEW 2011)
March 30 or 31, 2011, Essen, Germany

to be held at the 17th International Working Conference on
Requirements Engineering: Foundation for Software Quality (RefsQ'11)

Goal of the Workshop:

Most requirements engineering research so far has focused on specification quality, while ignoring practitioners that look for efficiency and pragmatism. The International Requirements Engineering Efficiency Workshop (REEW 2011) aims at initiating, facilitating, and nurturing the discussion on efficient approaches to engineer just good-enough requirements. Requirements engineering is here seen as a means that can be simplified, automated, or combined with other practices to achieve successful systems in an economically efficient manner. REEW 2011 will provide a platform for the community of practitioners and research experts that are interested in efficient and pragmatic approaches to requirements engineering.

Submission:

We invite original submissions in following formats:
- Full research papers (technical solutions and empirical studies, up to 12 pages)
- Full industry papers (experience reports, up to 12 pages)
- Short papers (vision, research previews and problem statements, up to 6 pages)

Important Dates:

Paper Submission: January 24, 2011
Notification of Acceptance: February 28, 2011
Camera-Ready Submission: March 14, 2011
REEW Workshop Date: March 30 or 31, 2011

Topics of Interest:

REEW 2011 will be dedicated to observations, concepts and approaches that allow understanding, facilitating, and increasing requirements engineering efficiency. Themes of interest for paper submission include, but are not limited to:

- Ad-hoc requirements engineering
- Pragmatic requirements-based collaboration
- Lean requirements specification
- Risk- and value-oriented requirements quality assurance
- Agile requirements management
- Requirements engineering process efficiency
- Efficient product family management

Program Committee

- Steffan Biffl, Technische Universität Wien (Austria)
- Vicenzo Gervasi, University of Pisa (Italy)
- Tony Gorschek, Blekinge Institute of Technology (Sweden)
- Paul Grünbacher, Johannes Kepler University (Austria)
- Andrea Herrmann, Axivion GmbH (Germany)
- Sven Krause, Zuehlke Engineering AG (Switzerland)
- Soo Ling Lim, University College London (U.K.)
- Nazim Madhavji, University of Western Ontario (Canada)
- Anna Perini, Fundazione Bruno Kessler (Italy)
- Zornitza Racheva Bakalova, University of Twente (Netherlands)
- Kurt Schneider, Leibniz Universität Hannover (Germany)
- Ken Schwaber, Scrum.org (U.S.A.)
- Inge van de Weerd, Utrecht University (Netherlands)

Organizers

Samuel Fricker
University of Zurich and Fuchs-Informatik AG (Switzerland)
fricker@ifi.uzh.ch

Norbert Seyff
University of Zurich (Switzerland)
seyff@ifi.uzh.ch

http://sites.google.com/site/reew2011/

Workshop

"Requirements Prioritization for customer-oriented Software-Development" (RePriCo'11)

http://www.bwi.uni-stuttgart.de/reprico11

March 30/31, 2011, Essen, Germany

At the 17th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ2011)

Call for Papers

Goals of the workshop

The workshop serves as a platform for the presentation and discussion of new and innovative approaches to prioritization issues for requirements engineering. Topics can either carry out new research approaches with a fundamental theoretical background or best-practices from practitioners with a focus on customer-orientation.

A high rate of active participation and intense exchange of ideas and experiences shall 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.

Topics of the workshop

Suggested topics include but are not limited to the following:

- Identification and prioritization of stakeholder and customer groups
- Analysis and prioritization of customer needs
- Elicitation and prioritization of long-term requirements
- Methods to evaluate and prioritize functional and non-functional requirements
- Techniques to classify and categorize requirements
- Deduction and prioritization of solution-independent requirements
- Identification and benchmarking of effective solution alternatives
- Estimation of interdependency between solutions and/or requirements
- Consideration of quality aspects
- Success factors of requirements prioritization
- Situational factors influencing requirements prioritization
- Cultural aspects
- Conflict and consensus management
- Evaluation of the economy of solutions
- Cost estimation, Cost-benefit analysis, risk analysis
- Tools for prioritizing requirements

Submission

We invite original submissions in following formats:

- Full research papers (up to 12 pages), including solution papers, empirical studies, surveys, and comparative studies,
- Experience reports (up to 12 pages), describing positive and negative experiences,
- Vision papers (up to 6 pages) stating where the research in the field should be heading towards,
- Problem statements (up to 6 pages) describing open issues of practical or theoretical nature,
- Research previews (up to 6 pages) reporting on research results at a premature stage.

Please use for formatting the LNCS style. See Springer's "Information for LNCS Authors" (http://www.springer.com/computer/lncs?SGWID=0-164-6-793341-0) for detailed information.

Publication

Papers will be published in the ICB Research Report by the Institute for Computer Science and Business Information Systems, University of Duisburg-Essen.

Important Dates

Paper submission deadline:	**24th of January**
Notification of acceptance:	**1st of March**
Camera-ready:	**14th of March**

Information

Benedikt Krams
Keplerstr. 17
70174 Stuttgart, Germany
Tel.: +49 (0)711/685-82386
krams@wi.uni-stuttgart.de
http://www.bwi.uni-stuttgart.de/reprico11

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
Dr. PD Andrea Herrmann, Axivion GmbH, Germany
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
Priv. Doz. Dr.-Ing. Robert Refflinghaus, TU Dortmund University, 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

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

Call for Papers

Workshop on Empirical Research in Requirements Engineering: Challenges and Solutions

EPICAL

https://sites.google.com/site/epical2010/

Held at REFSQ 2011

Organizers:

Andrea Herrmann, Axivion GmbH, Germany, herrmann@axivion.com
Maya Daneva, University of Twente, Netherlands, m.daneva@utwente.nl

Objective of the workshop, targeted outcome, targeted audience, and themes

The primary goal of this workshop is to create a forum and a community to debate the need for, the value of, and the challenges in using empirical approaches to researching aspects of RE processes and products. The targeted outcomes are (1) a preliminary agenda for conducting empirical research in RE, and (2) a plan for setting up an online forum for exchange of ideas, research designs and research results within the RE community. EPICAL brings together practitioners and researchers to debate on the research methods suitable in RE, the criteria for judging RE research outcomes, and the implications of choosing particular research designs for the validity of the obtained results. We invite submissions about one of these four central themes:

1. How to do RE research of good quality: action research, explorative case studies, experiments, replication studies, validation studies? To what extents is empirical research in RE different? Is it more qualitative? How do we make sure it is aligned to stakeholders' objectives?
2. How to judge good RE practice? Where do good RE practices come from? What criteria a RE practice should meet so that it obtains the 'good practice' status?
3. What is the role of anecdotal evidence in creating RE knowledge about effectiveness of RE approaches in context: needs for, value of, and challenges in evaluating lessons learnt and practitioners' experiences.
4. What is the role of empirical evidence in creating RE knowledge? How do we judge what we know and what we do not know in RE? How do we aggregate empirical results?

Submission deadline: January 24, 2011
Author Notification: March 1, 2011
Camera Ready Papers: March 14, 2011
Workshop Date: Wednesday, March 30 or Thursday, March 31

Submissions

EPICAL invites submissions in the form of experience reports and research papers. We encourage both practitioners and researchers to submit papers on their experiences, respectively, in carrying out empirical studies, or in deriving lessons learnt from real-life projects. We accept papers that reflect on experiences in successful as well as unsuccessful projects. We welcome both **full papers** (max. 15 pages) and **position papers** (max. 9 pages). We also invite s**tudent session papers** (full paper or position paper).

Submissions should be formatted according to the Springer LNCS format. Instructions are available at http://www.springer.com/computer/lncs?SGWID=0-164-6-793341-0

Papers should be submitted in a PDF format by email to the workshop chairs.
The papers will be published in the REFSQ Workshop Proceedings as an ICB Research Report with an ISSN Number. At least one author for each accepted paper must register for the workshop and present the paper.

Program Committee[1]

D. Berry (U of Waterloo, Canada), D. Damian (U of Victoria, Canada), M. Daneva (U of Twente, Netherlands), O. Gotel (Independent Researcher, New York City, USA), A. Herrmann (Axivion GmbH, Germany), M. Niazi (Keele University, UK), B. Paech (U of Heidelberg, Germany), O. Pastor (U of Valencia, Spain), C. Salinesi (Université Paris 1 Panthéon Sorbonne), K. Schneider (Leibniz University Hannover, Germany), I. van der Weer (U of Utrecht, Netherlands), R. Wieringa (U of Twente, Netherlands)

[1] The PC is in the process of being confirmed. Potentially more people can be added to the PC.

3 Requirements Engineering Efficiency Workshop (REEW)

Editors

Samuel Fricker
Blekinge Institute of Technology, Sweden, samuel.a.fricker@gmail.com

Norbert Seyff
University of Zurich, Switzerland, seyff@ifi.uzh.ch

Technical Programme

Introduction to the 1st International Requirements Engineering Efficiency Workshop (REEW 2011) at REFSQ 2011

Samuel Fricker[1] and Norbert Seyff[2]

[1] Blekinge Institute of Technology, Sweden
samuel.a.fricker@gmail.com
[2] University of Zurich, Switzerland
seyff@ifi.uzh.ch

Preface

This volume contains the papers presented at the 1st International Requirements Engineering Efficiency Workshop (REEW 2011) held on March 31, 2011 in Essen, Germany.

Requirements engineering research has for a long time focused on specification quality, leading to recommendations of how to engineer perfect requirements specifications. Practitioners, however, do not have the time, resources, and interests for overdoing requirements engineering. Rather, many situations call for short-cuts that allow investing effort in those concerns that are critical for success, while reducing effort in other areas where risk is comparably smaller. The social context, smart collaboration processes, and novel ways of looking at the interface between stakeholders and the supplier can be a basis to increase the yield of requirements engineering, while reducing required effort.

The International Requirements Engineering Efficiency Workshop (REEW 2011) aims at initiating, facilitating, and nurturing the discussion on efficient approaches to engineer just good-enough requirements. Requirements engineering is here seen as a means that can be simplified, automated, or combined with other practices to achieve successful systems in an economically efficient manner. REEW 2011 will provide a platform for the community of practitioners and research experts that are interested in efficient and pragmatic approaches to requirements engineering.

The REEW 2011 workshop program featured three talks on requirements engineering efficiency tactics and three talks on requirements engineering efficiency practice. Each submission was reviewed by 3 program committee members. We would like to thank the program committee for their timely reviews that provided valuable feedback to the authors.

March 16, 2011 — Samuel Fricker
Karlskrona and Zurich — Norbert Seyff

Program Committee

Stefan Biffl	TU Wien
Samuel Fricker	Blekinge Institute of Technology
Vicenzo Gervasi	University of Pisa
Tony Gorschek	Blekinge Institute of Technology
Paul Gruenbacher	Johannes Kepler University Linz
Andrea Herrmann	Axivion GmbH
Sven Krause	Zühlke Engineering AG
Soo Ling Lim	University College London
Nazim Madhavji	University of Western Ontario
Anna Perini	Fondazione Bruno Kessler
Zornitza Racheva	University of Twente
Juha Savolainen	Nokia Research Center
Kurt Schneider	Leibniz Universität Hannover
Ken Schwaber	Scrum.org
Norbert Seyff	University of Zurich
Inge Van De Weerd	Utrecht University

Faster from Requirements Documents to System Models: Interactive Semi-Automatic Translation

Leonid Kof, Birgit Penzenstadler

Fakultät für Informatik, Technische Universität München,
Boltzmannstr. 3, D-85748, Garching bei München, Germany
kof|penzenst@informatik.tu-muenchen.de

Abstract. [**Motivation:**] Natural language is the main presentation means in industrial requirements documents. This leads to the fact that requirements documents are often incomplete and inconsistent. Furthermore, if system models are constructed on the basis of these documents, explicit traces between the documents and the models are mostly missing.
[**Problem:**] Despite the fact that documents are mostly written in natural language, natural language processing (NLP) is barely used in industrial requirements engineering. Trade-offs of the existing NLP approaches hamper their broad usage: existing approaches either introduce a restricted language and, correspondingly, are able to process solely this restricted language, or, in the case of a non-restricted language, they cannot adapt to different writing styles, often co-existent in a single requirements document.
[**Principal ideas:**] Efficiency of system model construction and tracing can be improved by automated translation. Valuable side effects are early verification of requirements documents as well as tracing between the textual document and the constructed model.
[**Contribution:**] The presented paper shows how an NLP approach can be integrated in a CASE tool. The tool learns on the fly which grammatical construction represents which model element and enables text-to-model translation without restricting the allowed document language. The applicability of the tool in several case studies points towards improved efficiency for requirements analysis and design by using model generation from natural language requirements documents.

Keywords: requirements analysis, system modeling, natural language processing, automated generation, training

1 Requirements Documents Suffer from Missing Information

At the beginning of every software project, some kind of requirements document is usually written. The majority of these documents are written in natural language, as the survey by Mich et al. shows [1]. This results in the fact that the requirements documents are imprecise, incomplete, and inconsistent, because precision, completeness and consistency are extremely difficult to achieve using mere natural language as the main presentation means.

It is one of the goals of requirements analysis, to find and to correct the deficiencies of requirements documents. A practical way to detect errors in requirements documents is to convert informal specifications to system models. In this case, errors in documents would lead to inconsistencies or omissions in models, and, due to more formal nature of models, inconsistencies and omissions are easier to detect in models than in textual documents.

Despite many years of research in natural language processing (NLP), the transition from textual documents to system models is still performed manually in industrial praxis. This is due to the fact that existing NLP approaches aiming at text-to-model translation introduce restricted languages and can process solely texts written according to the restrictions. So far, restricted languages are barely accepted in industry. Another class of NLP approaches do not assume any language restrictions. However, these approaches do not extract complete models from the text, they mostly extract just lists of terms used in the document. Furthermore, they cannot adapt to different writing styles, often co-existent in a single requirements document.

Contribution The paper shows how an NLP approach can be integrated in a CASE tool and enriched with machine learning techniques, so that the integrated system provides text-to-model translation without constraining the allowed natural language.

2 Interactive Text-to-Model Translation

This section presents the tool concepts, the training data collection, the semi-automatic system model extraction procedure, and the evaluation of the tool.

2.1 Motivation: Statistical Machine Translation

The text-to-model translator presented in this paper is motivated by statistical machine translation and other statistical techniques for natural language processing. The basic idea of statistical machine translation [2] is to subdivide the translation process into two phases, namely training and actual translation.

The same basic idea is applied for statistical part-of-speech (POS) tagging [3] and parsing [4]: First, the tagger/parser takes a set of manually tagged/parsed sentences, as for example the Penn Treebank (`http://www.cis.upenn.edu/~treebank/`), and gathers statistics on POS tags or parse trees constituents. Then, given the statistical data and an input sentence, the tagger/parser produces the most probable sequence of POS tags, resp. the most probable parse tree for the given sentence.

We decided to integrate the collection of the training data, the training, and the actual text-to-model translation in one tool as there is no sufficiently sized training database available yet. The integration allows to gather document-specific training data, and to train the translator on these document-specific data. Thus, in our tool, the text-to-model translation consists of manual modeling and automatic extraction of model elements. Manual modeling is used to gather the training data, whereas automatic extraction analyzes the training data and proposes new model elements.

The details of the training data collection, training, and actual translation are presented below: Section 2.2 presents the training data collection, and Sections 2.3 and 2.4 present the training and actual text-to-model translation for different types of model elements.

2.2 Training Data Collection: Manual Modeling

The goal of manual modeling, considered as a part of text-to-model translation, is to gather training data for the subsequent automatic translation. From the point of view of user interaction, the tool distinguishes two types of model elements:

- entities, characterized solely by their names, and
- relations, characterized both by their names and the entities involved in the relation.

For example, a state of an automaton is an entity, as it is characterized solely by its name, but a state transition is a relation, as it is characterized by its name, and two states involved in the transition.

To create a model element, the tool user has to mark a word sequence in the text field showing the requirements document, and then to select the model element type from the context menu. The marked word sequence becomes the name of the newly created model element.

When creating a new model element, the tool creates an explicit trace between the model element and the previously marked word sequence. Every trace specifies the sentence, the exact position of the word sequence used to create the model element name, and the model element itself. These traces are used later as training data items for the automatic text-to-model translation.

2.3 Automatic Extraction of Entities

From the point of view of the tool user, extraction of entities looks similar to the manual creation of model elements: the user selects a document section, and then chooses from the context menu, which model element type should be extracted. Then, the tool proposes the names of potential new model elements to the user. The user selects, which names should be really used to create new model elements.

Internally, the tool keeps track of the user decision and augments the training data: for every model element name selected by the user, the tool creates a trace from the name occurrence in the text to the newly created model element, and makes the created trace to a positive example (new training data item). Every not selected name becomes a negative example and is not presented to the user again. The tool does not distinguish between manually created and automatically extracted model elements and puts them in the same sets of positive/negative examples, as shown in Figure 1. The fact that the tool does not distinguish between manually created and automatically extracted model elements implies an important advantage: the tool does not force the user to create all the training data first, and then to perform automatic extraction only. Instead, the tool allows for arbitrary interleaving of manual modeling and automatic extraction steps.

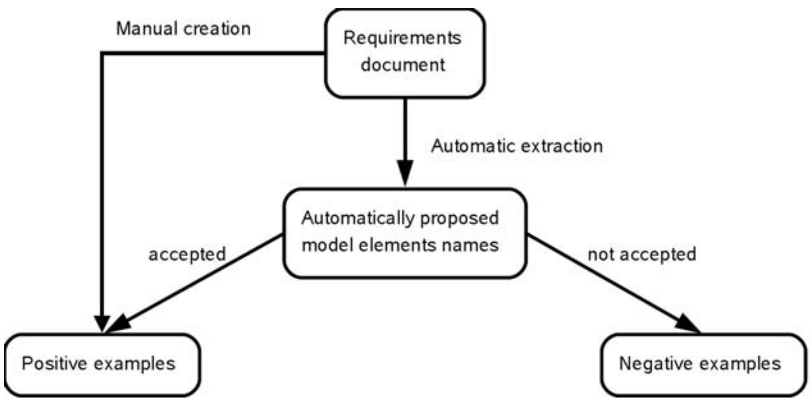


Fig. 1. Training data management

For automatic extraction of model element names, the tool performs the following steps:

1. **Trace detection:** turn every name occurrence into a training data item.
2. **Sentence properties analysis:** gather statistics on types of sentences with traces.
3. **Heuristics initialization:** decide on most suitable heuristics for trace.
4. **Heuristics application:** provide set of term extractions, present to user.

The extraction steps are presented below in detail.

Trace detection Before starting the extraction of model elements names, the user has to select a document section. The rationale for this design decision is the fact that industrial requirements documents are often written by different authors, using different writing styles. Both the learning procedures and the extraction of model elements names are more efficient when applied to a homogeneously written document section.

When the user selects the document section and the model element type to be extracted, the tool searches for the traces of the selected model element type in the selected document section. The rationale is that the names of different model element types can be linguistically different, so the learning procedure is always applied only to the traces of the model element type selected by the user.

Training data gathered in this way are used in the subsequent learning and extraction steps.

Sentence properties analysis In order to handle compound sentences, the tool uses the information provided by the parser and splits every compound sentence into elementary parts (segments), cf. [6]. A segment is defined as a sentence part containing at most one verb phrase. Then, the tool gathers statistics regarding the properties of sentence segments containing traces.

The tool gathers statistics on three properties of the sentence segments containing traces: (1) segment contains "if" or "when", (2) segment contains passive voice, and (3) main verb of the segment. The first two properties are binary: the tool counts the positive/negative examples contained in the sentence segments having the analyzed property. If the difference between the positive and negative examples exceeds the predefined threshold, the tool activates the corresponding property filter. Activated property filters are used for the extraction of new model element names. For the last property (main verb), the tool picks the verb used in the most sentence segments containing positive examples. Verbs are identified by means of part-of-speech tags provided by an external tagger.

As a first approximation, the tool extracts new model elements names from sentence segments having the same properties as the sentence segments containing the names of the previously created model elements.

Heuristics initialization In order to extract model elements names, the tool provides a set of heuristics. Most extraction heuristics are based either on the part-of-speech tags or simply on the sentence structure:

- Heuristics based on the work by Hearst [7]: The tool looks for two special sentence structures, namely "X, such as A, B, and C", and "X ⟨some-text⟩: A, B, and C". In both cases, A, B, and C can be extracted.
- The tool can extract verbs with their prepositions.
- The tool can extract whole sentence segments.
- Named entity recognition (NER): A named entity, in the context of the presented method, consists of a keyword, followed by a sequence of nouns or adjectives ("mode emergency stop", "mode initialization",. . .) The keyword can also be the last word of the named entity ("emergency stop mode", "initialization mode"). The keyword for named entity recognition is derived from the training data.

In order to determine, which heuristics should be used for the actual extraction of new model elements names, the tool determines the motivating heuristic for every trace occurring in the training data. To determine the motivating heuristic for a given trace, the tool sequentially applies all available heuristics to the traced sentence, and picks the heuristic whose extraction result is most close to the traced word sequence. For example, "normal mode" in Sentence 1

normal mode

1. the normal mode is the standard operating mode in which the program tries to maintain the water level in the steam-boiler between n1 and n2 with all physical units operating correctly.
2. as soon as the water level is below n1 or above n2 the level can be adjusted by the program by switching the pumps on or off.
3. as soon as the program recognizes a failure of the water level measuring unit it goes into rescue mode.
4. failure of any other physical unit puts the program into degraded mode.
5. if the water level is risking to reach one of the limit values m1 or m2 the program enters the mode emergency stop.
6. this risk is evaluated on the basis of a maximal behaviour of the physical units.
7. a transmission failure puts the program into emergency stop mode.

Table 1. Steam Boiler Specification [5], excerpt

from Table 1 can be extracted with two heuristics: "NER" with keyword "mode" and "sentence subject". In order to determine the motivating heuristic in such a case where the same word sequence can be extracted with several heuristics, the tool applies a priority list and picks the heuristic with the highest priority. The priority list results from our previous work [8]: the better results a heuristic provided in our previous studies, the higher its priority:

$$\begin{aligned} &priority(NER) > priority(verb + preposition) > \\ &priority(Hearst) > priority(subject/object) > \\ &priority(before\ subject/after\ object) > \\ &priority(whole\ sentence\ segment) \end{aligned}$$

Every heuristic that motivates some word sequence contained in the training data, and, in the case of several heuristics motivating the same word sequence, has higher priority than its competitors, is used to extract new model elements names.

Heuristics application To extract new model element names, every heuristic motivating a certain training data item is applied to the whole document section selected by the user. In order not to annoy the user with already accepted or already rejected model elements names, the tool purges the positive and negative examples from the set of the extracted model elements names (cf. Figure 1). In order not to overwhelm the user with too many new model elements names, the tool filters the sentences that are used for the extraction: First, the tool extracts new model element names exclusively from sentences with the same properties as determined above (see paragraph "Sentence properties analysis").

If the number of newly extracted model elements lies below the predefined threshold, the tool successively attenuates the sentence property filter. Attenuation of binary properties ("if"-containment and passive voice) means that the particular property filter is simply turned off. Attenuation of the verb property takes place in two steps: for the fully activated property filter, the tool picks the sentence segments containing the particular verb. After the first attenuation step, the tool picks the sentence segments containing some verb. After the second attenuation step, the property filter is turned off. As soon as the number of newly extracted model elements rises above the threshold or all the property filters are turned off, the attenuation and the extraction procedure terminate.

The tool presents the extraction results in a dialog (see Figure 2). In order to make the extraction transparent, the tool groups the extracted model elements names by the heuristic used for the extraction, and presents the heuristic and the used property filters together with the extracted model elements names. Then, the user has to select, which of the proposed model elements become real model elements. This feedback procedure augments both the model and the training data, as shown in Figure 1.

Currently, the extraction tool supports three types of entities: components, states, and MSC (Message Sequence Charts) actors.

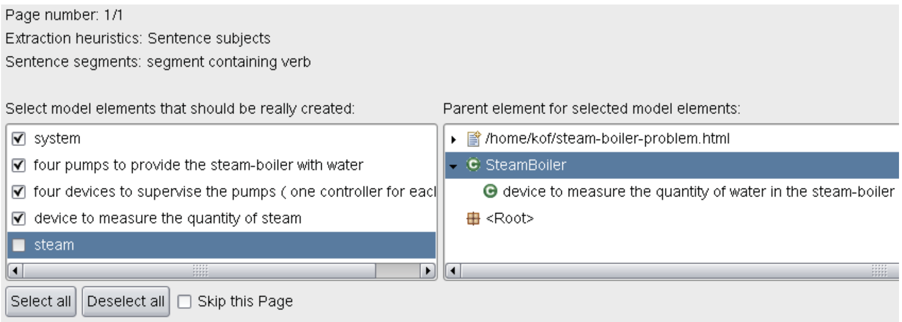


Fig. 2. Automatic extraction of model elements, selection of new model elements to be created and a container

2.4 Automatic Extraction of Relations

In order to extract relations (for example, state transitions), it is necessary not only to extract the name of the relation, but also to determine the entities involved in the relation. (E.g., in the case of state transitions, it is necessary to determine the involved states.) For proper identification of entities involved in a relation, a discourse model is necessary: For example, Sentence 5 from Table 1 states that the system switches from "normal mode" to "emergency stop mode" under certain circumstances. Although "normal mode" does not occur in Sentence 5, it is perfectly obvious for the human reader that "normal mode" is the source state of the transition specified in Sentence 5.

In order to find the entities involved in a relation, the tool uses a simple discourse model, similar to the model that was proven to be useful in our previous work [9].

In order to extract relations, the tool extracts relation names first, using the procedures presented in Section 2.3. Then, for every relation name that was accepted by the user, the tool looks for entities potentially involved in the relation: It follows text paths obtained from the training data and searches for the names of entities already existing in the model. For example, if Sentence 5 from Table 1 is the only training data item for state transitions, the tool will always look for the source state in the header of the section containing the name of the extracted state transition, and for the target state in the same sentence as the name of the extracted state transition.

The found entities, potentially involved in the relation, are presented to the user. The user selects, which entities are really involved in the relation, and after that the tool creates the relation, using the extracted relation name and the selected entities.

Currently, the extraction tool supports four types of relations: channels, state transitions, MSC messages, and MSC assertions (no explicitly involved entities).

3 Evaluation

The tool was evaluated on four specifications with respect to applicability: Steam Boiler [5], Autopilot [10], Bay Area Rapid Transit (BART) [11], and Instrument Cluster [12]. B The goal of the evaluation was to investigate if the extraction results keep up with the expectations of the tool author. Due to possible evaluator's bias, no further evaluation was performed. The remainder of this section presents the evaluation results for the four specifications used for the evaluation.

Steam Boiler. This specification describes the steam boiler itself and states the requirements to the control program for the steam boiler. The goal of the control program is to maintain the water level, even if certain equipment fails, between predefined marks, in order to prevent damage of the steam boiler. To construct the model, "steam boiler" was manually marked as a component. The tool managed to extract the remaining components automatically. Then, "initialization" (mode) was manually marked as a program state, and the tool extracted all other operating modes. To extract state transitions, one "if" sentence, similar to the Sentence 5 from Table 1, was used as a training data item. This enabled the tool to correctly extract all state transitions encoded in "if"-sentences. In the second extraction step, the tool attenuated the sentence property filter and was able to extract state transitions like "as soon as ..." (Sentence 3 from Table 1 and similar), as well as "failure of ... leads to ..." (Sentence 4 from Table 1 and similar). Finally, Sentence 7 from Table 1 was manually marked as a state transition. The tool extracted similar sentences as state transitions, but also produced some noise, that had to be filtered out manually in dialogs like the dialog shown in Figure 2.

Autopilot. The autopilot case study is extremely short (just 2 pages of requirements) and describes the user interface (buttons and displays) and the operating modes of an autopilot system. Two operating modes had to be manually marked as states, then the tool extracted the remaining modes. Similarly, one of the displays was marked as a component, and the tool extracted the other displays as components. As the specification does not explicitly specify state transitions or connections between components, no further extraction was performed.

BART. The BART (Bay Area Rapid Transit) specification describes the train control system itself and safety requirements that must be satisfied by the control system. To model the BART system, “non-vital station computer” was manually marked as a component, and the tool extracted other controllers (“vital station computer”, “station computer”) as components. Furthermore, as the specification contains precise information on different phases of the braking process, these informations were modeled too. Here, we used mock up modeling: we manually marked the name of one of the braking phases as a component, and the tool was able to extract the other breaking phases. The resulting model does not really make sense, but it allowed us to evaluate the linguistic part of the tool. As the specification does not explicitly specify state transitions or connections between components, no further extraction was performed.

Instrument cluster. The Instrument Cluster specification describes the optical design of the instrument cluster as a part of the car dashboard, its hardware, and, most importantly, its behavior. Behavior specification consists of 10 use cases, with several scenarios for every use case. To translate scenarios to message sequence charts, “driver” and “car” were manually marked as components, then the tool was able to extract further components involved in scenarios (“instrument cluster”, “ignition”, “motor”, . . .). After the extraction of components, “switches on”, taken form the sentence “The driver switches on the car”, was manually marked as an MSC message. Then, the tool automatically extracted further messages (verbs from different sentences). To summarize, in all case studies, the tool was able to keep up with the expectations of the tool author.

4 Related Work

Work related to the presented paper can be subdivided in two areas: work on text-based modeling and work on natural language processing (NLP) in requirements engineering.

4.1 Text-Based Modeling

Saeki at al. [13], Overmyer et al. [14], and Ermagan et al. [15] introduced tools providing modeling approaches related to the approach presented in this paper. The approach by Saeki et al. allows the user to mark words in the requirements documents, and then assign them to a word type. Then, the approach maps nouns to classes (object-orientated), and verbs to operations, but can handle just four predefined verb classes and is not able to learn on the fly, as the approach presented in this paper. Overmyer et al. developed a tool allowing the user to mark words or sequences and map them to classes, roles, and operations. They do not assume that the verb must fall into one of the predefined categories. Nevertheless, the tool by Overmyer et al. cannot learn either. Ermagan et al. developed the tool SODA, allowing to link textual use cases to behavior models. Such links are similar to traces introduced in the presented paper, but SODA sticks to manual modeling.

4.2 Natural Language Processing in Requirements Engineering

Natural language processing is applied in assessment of document quality, identification and classification of application specific concepts, and analysis of system behavior.

Assessment of document quality. Rupp [16], Fabbrini et al. [17], Kamsties et al. [18], and Chantree et al. [19] define writing guidelines and measure their satisfaction. Their aim is to detect poor phrasing and to improve it; they do not target at system modeling, as our approach.

Identification of application specific concepts. Goldin and Berry [20], Abbott [21], or Sawyer et al. [22] analyze the requirements documents, extract application specific concepts, and provide an initial static model of the application domain. However, these approaches cannot learn on the fly which concepts should be extracted, and they do not perform any behavior modeling.

Analyzing system behavior. Vadera and Meziane [23], Gervasi and Zowghi [24], Breaux et al. [25, 26], and Avrunin et al. [27] translate requirements documents to executable models by analyzing linguistic patterns. Vadera and Meziane propose a procedure to translate certain linguistic patterns into first order logic and then to the specification language VDM, but they do not provide automation for this procedure. Gervasi and Zowghi go further and introduce a restricted language, a subset of English. They automatically translate textual requirements written in this restricted language to first order logic. Similarly, Breaux et al. introduce a restricted language and translate this language to description logic. The approach by Avrunin et al. translates natural language to temporal logic.

Our work goes further than the above approaches, as we do not assume or enforce language restrictions. Treatment of non-restricted language is possible due to the ability of the presented approach to learn on the fly.

To summarize, to the best of our knowledge, there is no approach to documents analysis, yet, able to translate model descriptions written in non-restricted natural language to models themselves, and to learn on the fly which phrases should be translated to which part of the model.

5 Discussion and Conclusion

Both requirements engineering and system modeling are non-trivial tasks and the presented approach does not claim to solve all their problems. However, it provides important links between textual requirements and modeling:

- **Efficient analysis and early construction:** Given a textual requirements document, it helps to explore the document and to construct a system model.
- **Automated tracing:** When constructing the model, it maintains explicit traces between the model and the textual document.
- **Early verification of requirements documents:** If the model description in the document is incomplete, it makes this incompleteness apparent, by creating an incomplete system model.

In this way, the presented approach closes the gap between textual specifications and models in an efficient way to support lean and fast requirements engineering.

Benefits. The main benefit of the integration of an NLP approach is not the extracted model itself, but the extraction procedure: by extracting both model elements and relations between them, NLP would contribute to thorough exploration and thereby early verification of the requirements document. These corrections can also result in the necessity to change the specification text. Thus, the CASE tool with integrated linguistic techniques provides means for tracing model elements back to the specification text. Although a specific tool was used to demonstrate the main idea (AutoFOCUS `http://af3.in.tum.de/`), the linguistic part of the approach remains tool-independent.

Limitations. Complete translation does not mean that the resulting model can be directly used for system simulation or code generation. Both due to inconsistencies and omissions in the requirements document, and due to imperfection of the linguistic tools, the resulting model will most probably need corrections, before it can be used for system simulation or code generation. Nevertheless, the partial automation and the integration of requirements analysis and early design improve the efficiency of requirements engineering processes.

Outlook. The linguistic part of the approach presented in this paper is generic: it does not depend on the CASE tool, nor on the types of model elements or relations that should be extracted. This implies that the presented approach to text-to-model translation can be integrated into every CASE tool, thus allowing us to translate a requirements document to the model most appropriate for the particular application domain. This, in turn, ensures industrial applicability of the presented approach.

References

1. Mich et al.: Market research on requirements analysis using linguistic tools. Requirements Engineering **9** (2004)
2. Koehn, P.: Statistical Machine Translation. Cambridge University Press (2010)
3. Curran et al.: Multi-tagging for lexicalized-grammar parsing. In: 21st International Conference on Computational Linguistics. (2006)
4. Clark & Curran: Parsing the WSJ using CCG and log-linear models. In: ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics. (2004)
5. Abrial et al.: The steam boiler case study: Competition of formal program specification and development methods. In: Formal Methods for Industrial Applications. (1996)

6. Kof, L.: Treatment of Passive Voice and Conjunctions in Use Case Documents. In: Application of Natural Language to Information Systems. (2007)
7. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th conference on Computational linguistics. (1992)
8. Kof, L.: Requirements Analysis: Concept Extraction and Translation of Textual Specifications to Executable Models. In: Application of Natural Language to Information Systems. (2009)
9. Kof, L.: Translation of Textual Specifications to Automata by Means of Discourse Context Modeling. In: Requirements Engineering: Foundation for Software Quality, 15th International Working Conference. (2009)
10. Butler, R.W.: An introduction to requirements capture using PVS: Specification of a simple autopilot. Technical report, NASA Langley Research Center (1996)
11. Winter et al.: The BART case study. In: Formal Methods for Embedded Distributed Systems. (2004)
12. Buhr et al.: DaimlerChrysler demonstrator: System specification instrument cluster. Project Empress ITEA (2004)
13. Saeki et al.: Software development process from natural language specification. In: Proc. of the 11th Intl. Conf. on Software Engineering. (1989)
14. Overmyer et al.: Conceptual modeling through linguistic analysis using LIDA. In: Proc. of the 23rd Intl. Conf. on Software Engineering. (2001)
15. Ermagan et al.: Towards Tool Support for Service-Oriented Development of Embedded Automotive Systems. In: Proceedings of the Dagstuhl Workshop on Model-Based Development of Embedded Systems (MBEES'07). (2007)
16. Rupp, C.: Requirements-Engineering und -Management. Professionelle, iterative Anforderungsanalyse für die Praxis. Hanser-Verlag (2002)
17. Fabbrini et al.: The linguistic approach to the natural language requirements quality: benefit of the use of an automatic tool. In: 26th Annual NASA Goddard Software Engineering Workshop. (2001) `http://fmt.isti.cnr.it/WEBPAPER/fabbrini_nlrquality.pdf`, accessed 08.02.2010.
18. Kamsties et al.: Detecting ambiguities in requirements documents using inspections. In: Workshop on Inspections in Software Engineering. (2001)
19. Chantree et al.: Identifying nocuous ambiguities in natural language requirements. In: Proc. of the 14th IEEE Intl. Requirements Engineering Conference. (2006)
20. Goldin & Berry: AbstFinder, a prototype natural language text abstraction finder for use in requirements elicitation. Automated Software Eng. **4** (1997)
21. Abbott, R.J.: Program design by informal English descriptions. Communications of the ACM **26** (1983) 882–894
22. Sawyer et al.: Shallow knowledge as an aid to deep understanding in early phase requirements engineering. IEEE Trans. Softw. Eng. **31** (2005)
23. Vadera, S., Meziane, F.: From English to formal specifications. The Computer Journal **37** (1994) 753–763
24. Gervasi, V., Zowghi, D.: Reasoning about inconsistencies in natural language requirements. ACM Trans. Softw. Eng. Methodol. **14** (2005) 277–330
25. Kiyavitskaya et al.: Automating the extraction of rights and obligations for regulatory compliance. In LNCS, ed.: ER. Volume 5232. (2008)
26. Breaux et al.: Semantic parameterization: A process for modeling domain descriptions. ACM Trans. Softw. Eng. Methodol. **18** (2008)
27. Smith et al.: PROPEL: an approach supporting property elucidation. In: Proceedings of the 24th International Conference on Software Engineering. (2002)

Efficiency in Requirements Engineering through Feed-forward

Sharmin Ahmed, Muhammad Iftekher Chowdhury,
Remo Ferrari and Nazim H. Madhavji

Department of Computer Science, University of Western Ontario,
London, Canada, N6A 5B7
{sharmin.ahmd, iftekher.chowdhury, ferrari.remo, madhavji}@gmail.com

Abstract. There are different ways in which the efficiency improvement of a requirements engineering (RE) process can be addressed. Examples are: automation through tool support, reducing peripheral documentation, reducing the number and types of artefacts processed, detailing requirements only when needed, etc. While such approaches may at first seem to increase the efficiency, there may well be negative long-term consequences, for example, in terms of increased backtracking and rework, lower quality of downstream artefacts and the eventual product, delayed delivery of the system, etc. Such consequences counteract the initial gains in RE. In this vision-category paper, we present a new idea, that of a feed-forward (FF) centric RE process. FF is an "anticipatory activity" of passing information from one location in the RE process to another, where the recipient of the information could possibly benefit by making use of it "ahead of time". This could result in savings of time; avoidance of unnecessary activities; execution of necessary, but as yet not considered, activities; etc., all leading to improved RE decisions. While FF is known in other domains such as systems engineering and management, to our knowledge, it has not been scientifically investigated or formulated as a vision in RE. In this paper, we discuss the concept of FF in terms of the five Ws and one H (i.e., why, what, where, when, who, and how) and also present example requirements for enabling FF in an RE process.

Keywords: Requirements Engineering, Software process improvement, Feed-forward, Software team communication, agile processes.

1 Introduction

There are different ways in which the efficiency improvement of a requirements engineering (RE) process can be addressed. Example approaches include: automation through tool support [16], reducing peripheral documentation [6], reducing the number and types of artefacts processed [6], detailing requirements only when needed [6], etc. While such approaches may at first seem to increase RE efficiency, there may well be negative long-term consequences. For example, a study involving 16 organisations practising agile approaches shows that the quality of downstream artefacts is reduced if less emphasis is placed on security and performance requirements early in the RE process, ultimately resulting in increased backtracking and rework [4]. Similarly, research described in [9] shows that reduced front-end documentation leads to information loss which, in turn, can have a negative impact on the product and process quality over time. Furthermore, though locally focused tools (e.g., RE-only tools) may lead to improved local artefacts and processes, they can also lead to fragmentation across different life-cycle processes (e.g., architecting) because of the lack of tool integration across such processes, which can limit developer capabilities and productivity [15]. Such consequences counteract the initial efficiency gains in RE.

In this vision-category paper, we present an orthogonal idea of a “feed-forward” (FF) centric process for improving efficiency in RE. FF is defined in non-software literature as: “ … some output of an earlier step is fed into a step occurring down the line” [7]. For software projects, we interpret FF as: “an anticipatory activity of passing information from one location in the process to another where the information could possibly be used ahead of time”. Note here that “ahead of time” is quite compatible with the idea of “down the line” in the definition and has a direct impact on process efficiency. Also, the FF information need not be intra-process, meaning that the source of information could lie outside the affected process [5]. Contrast this with the “normal flow” of the development process; information forwarding is not anticipatory and is not passed ahead of time. Feed forwarding information under favourable conditions can aid recipients to get on with their tasks and decision-making, possibly even altering the course of action they would take because of the FF information they have received.

The notion of FF is already being used in other domains such as systems engineering [13] and management [10]. However, in RE and software engineering (SE) practice, the concept of FF is not engrained in the process (contrast this with the concept of “feedback”); there are only anecdotal references to it [1], [8], [15].

Our vision of FF in RE is particularly motivated from our observations from a previous empirical study [15]. We postulated that if architects had access to critical RE information prior to the validation of the full set of requirements, then this FF information could trigger early groundwork for specific architectural enhancements, while other requirements were still being elicited. Such agile use of FF information has potentially significant benefits for RE process efficiency, in particular: effort minimization; early highlighting of implementation risks; minimising effort expended; ensuring a tighter coordination between varying project roles; etc.

Note that these benefits and usage of FF complement those from other RE efficiency approaches (for example, process improvement [17], automation [16], etc.). FF is not meant to supplant these existing approaches but can be used in conjunction to increase RE efficiency.

In the next section, we describe related work, following which, in Section 3, we discuss the concept of FF in terms of the five Ws and one H (i.e., why, what, where, when, who, and how). In Section 4, we give example requirements for enabling FF in an RE process. Finally, Section 5 summarises the paper and mentions our on-going work.

2 Background

In this section, we first overview examples of FF from other disciplines (control engineering, artificial neural networks, and management) before describing examples of FF in SE.

A well-known application of FF is in the domain control engineering [13], where FF in control systems are “open loop” systems where corrective action is taken *before* measuring and acting upon the output of the system. This is in contrast to “closed loop” systems where corrective action (feedback) is taken *after* an error is detected in the output and is reported back to the source. The thermostat of an house is an example [5] of a closed loop feedback controller. This controller relies on measuring the controlled variable (the temperature of the house), and then adjusting the output (to turn on/off the heater). However, feedback control usually results in periods of time where the controlled variable is not at the desired setpoint (i.e., the room temperature is below or above the set temperature on the thermostat). Thus, if the door of the house were opened on a cold day, the house would cool down. After it fell below the setpoint, the heater would turn on, but there would be a period when the house is colder than desired. FF control can avoid the slowness of feedback control. With FF control, the disturbances (i.e., events other than inside-house temperature) are measured and accounted for before the feedback controlled affects the system. Thus, a FF system may measure the fact that the door is opened and automatically switches on the heater before the house gets too cold.

Other examples of FF are in artificial neural networks [11], where pre-determined information is fed forward into the system and the parameters of the network are adjusted to minimise deviations from the expected output. In [10], Goldsmith discusses how a ten-minute exercise on FF practised by business professionals is preferable to feedback for day-to-day interactions. The reason is that feedback is often taken personally as a negative but FF does not suffer this fate.

In SE, we also observe examples of FF. In [1], Agresti mentions how anticipated staff-turnover in the forthcoming weeks - a piece of human-resource information that may not be known in a given software project - can be fed forward to the appropriate project manager(s) so that it can be incorporated into the time-estimation models to recalculate the project estimates and to assess risks due to staff turnover. In [8], Fricker et al., propose a goal-oriented systems model for the communication of requirements from product management to development teams. One paradigm described in this model explicitly mentions the use of FF, where the product manager uses their knowledge of the development teams’ expertise to anticipate the project output and accordingly adjust the requirements to be elicited. In [12], Lehman et al., mentions that software project and process issues (such as specification changes, unexpected conditions, performance problems) can be overcome through proper usage of a mixture of feedback and feed-forward in software process.

While FF is mentioned in SE [1], [12], [15] and RE [8], no scientific research on FF has been conducted to our knowledge. This paper is a first step to fill this gap.

3 The Concept of Feed-forward in the RE context

In this section, we analyse the concept of FF, in the context of RE, in terms of the five Ws and one H[1] (i.e., who, when, where, why, what, and how) to gain a better understanding of the underlying issues.

Who and When: Any project stakeholder (e.g., product manager, release manager, RE analyst, etc.) within a RE process can potentially feed information forward to another stakeholder. A key precondition for FF is that it must occur *ahead of time* of a given event in the process. For example, analysts can feed forward critical requirements information to the software architect during elicitation *prior* to requirements validation so that architect can take preparatory early action.

Where: The idea of FF mentioned in [15] is an example of "inter-process" FF, where information is fed forward, in this case, from the RE process to the software architecting process. FF can also occur "intra-process". For example, in a large distributed development effort, a sub-team refining the high-level requirements for their components could influence the (say, prioritising) decisions being made by another sub-team.

Why: In the Introduction section, we described the overall motivation and rationale behind incorporating FF in the RE process. Here, we list several additional reasons for practising FF in RE. This rationale is based on general management literature [10] but it can also apply to RE.

- FF tends to be faster and more efficient than feedback.
- We cannot change the past but we can change the future.
- It can be more productive to help people to be right, than to prove that they were wrong.
- FF is not taken as personally as feedback.
- Unlike FF, feedback can reinforce personal stereotyping and negative self-fulfilling prophecies.
- People do not like giving or receiving negative feedback. On the other hand, FF is always positive because it concerns the future.

What: Any type of information can potentially be fed forward (example: requirements, costs data, implementation difficulty and risk information, etc.).

How: Information can be fed forward using any possible means of communication used in the RE process, such as e-mail, face-to-face meetings, instant messaging, telephone, video-conferencing, memos and other forms of documentation. Teams or individuals may be co-located or distributed, and based on the distribution the format of the communications may vary. Also, FF information can be communicated indirectly through other people in the organisation, or more commonly referred to in the literature as information or knowledge brokers [3] [14].

---

[1] A well-known formula in journalism and research ([2], p.8-10) to cover a topic comprehensively.

4 Example Requirements

While the idea of FF is appealing, omnipresent evidence suggests that, unlike feedback, it is not a "needs-driven" activity. In the case of feedback, for instance, we know readily that the quality of the emerging requirements artefacts and of the eventual solution are going to suffer without integrating feedback into the process. Even a low amount of feedback is likely to yield tangible gains in quality. On the other hand, in the case of FF, the benefits are not as clear to everyone, though there is hope. Also, effective FF requires an elevated level of thinking and judgment from the stakeholders involved. In terms of maturity of the RE process, one can argue that "feedback" is necessary in low maturity processes; whereas, existence of cost-effective and fruitful FF in a process would suggest an incredibly high maturity of the process.

Here, we list several example requirements to enable FF in an organisation's RE process:

- For FF to be cost-effective in a project, individual stakeholders possessing FF information need to be familiar with other stakeholders' tasks and activities that may benefit from the FF information.
- Stakeholders need to be trained on the concept of FF and on the conditions when FF should (or could) be, or not be, used.
- Organisational motivation (e.g., through recognition, rewards and incentives) and self-motivation are key factors in institutionalising successful FF in an organisation.
- There is a need to make the cost-benefit ratio of FF, from release to release, visible to the stakeholders (a form of feedback!) so as to make improvements with respect to FF.
- FF requires a high degree of organisational and project maturity.

These requirements implicitly suggest FF-related principles to be engrained in the fabric of the organisation. We do not, for even one moment, assume that implementation of these requirements is trivial. However, prior to any attempt at implementing such requirements it is important that empirical studies are conducted to ascertain the cost-benefits of FF.

5 Summary and On-going Work

This short paper describes the idea of a feed-forward (FF) centric process with the goal of improving RE efficiency. FF is an "anticipatory activity" of passing information from one location in the RE process to another or to other SE processes, and vice versa, where the recipient of the information can benefit by making use of the information ahead of time. The paper discusses the concept of FF in terms of the five W's and one H (i.e., why, what, where, when, who, and how) [2]. Also presented are several example requirements (in Section 4) for enabling FF in an organisation's RE process. We do not consider successful implementation of such requirements to be trivial because it requires embracing the idea of FF at all levels in an organisation. However, prior to embarking upon organisational change, it is important to conduct empirical studies to assess the costs-benefits of FF. As part of our ongoing work on FF, we are conducting preliminary case studies and an industrial survey. These will be reported when the studies are complete. We hope that the ideas presented in this paper can act as a stepping-stone towards increased awareness of FF in the field of RE, eventually leading to the adoption of FF practices in industrial-scale RE processes.

References

1. Agresti,W.W.: A Feedforward Capability to Improve Software Reestimation. In: Madhavji, N. H. et al. (eds.) Software Evolution and Feedback, pp. 443--458. John Wiley & Sons, West Sussex, UK (2006)
2. Blaikie, N.: Designing Social Research: The Logic of Anticipation. Polity Press, Cambridge, UK (2000)
3. Boden, A., Avram, G.: Bridging knowledge distribution - The role of knowledge brokers in distributed software development teams. In: 31st International Conference on Software Engineering (ICSE'09) - Workshop on Cooperative and Human Aspects on Software Engineering, pp. 8--11. IEEE Computer Society, Washington, USA (2009)
4. Cao, L., Ramesh, B.: Agile Requirements Engineering Practices: An Empirical Study. IEEE Software. 25 (1), 60--67, IEEE Computer Society, CA, USA (2008)
5. Controller (control theory) - Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Controller_(control_theory)
6. Davis, A.: Just Enough Requirements Management: Where Software Development Meets Marketing. Dorset House Publication Co., Inc., New York, USA (2005)
7. Feed forward definition, http://www.businessdictionary.com/definition/feed-forward.html
8. Fricker, S., Gorschek, T., Glinz, M.: Goal-Oriented Requirements Communication in New Product Development. In: Second International Workshop on Software Product Management, pp. 27--34. IEEE Computer Society, CA, USA (2008)
9. George, B., Bohner, S. A., Prieto-Diaz, R.: Software information leaks: a complexity perspective. In: 9th International Conference on Engineering Complex Computer Systems (ICECCS'04), pp. 239--248. IEEE Computer Society, CA, USA (2004)
10. Goldsmith, M.: Try Feedforward Instead of Feedback. Leader to Leader. 25, 11--14, Leader to Leader Institute, NY, USA (2002)
11. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice-Hall, Englewood Cliffs, NJ, USA (1999)
12. Lehman, M. M., Ramil, J. F.: Software Evolution and Software Evolution Processes. Annals of Software Engineering. 14, 275—309, J. C. Baltzer AG, Science Publishers, Red Bank, NJ, USA (2002)
13. Leigh, J. R.: Control Theory. Institute of Engineering and Technology, UK (2004)
14. Marczak, S., Damian, D., Stege, U., Schroter, A.: Information Brokers in Requirement-Dependency Social Networks. In: 16th International Requirements Engineering Conference (RE '08), pp.53--62. IEEE Computer Society, CA, USA (2008)
15. Miller, J. A., Ferrari, R., Madhavji, N: An Exploratory Study of Architectural Effects on Requirements Decisions. Journal of System and Software. 83, 2441--2455, Elsevier Science Inc., NY, USA (2009)
16. Requirements Tools, http://www.volere.co.uk/tools.htm
17. Sommerville, I, Ransom, J.: An empirical study of industrial requirements engineering process assessment and improvement. ACM Transactions on Software Engineering Methodology (TOSEM). 14, 85--117, ACM, NY, USA (2005)

Towards Faster Application Engineering through Better Informed Elicitation – A Research Preview

Sebastian Adam

Fraunhofer IESE, Fraunhofer Platz 1, 67663 Kaiserslautern
sebastian.adam@iese.fraunhofer.de

Abstract. Application engineering (AE) is still a time-consuming and expensive task, and the benefits from using a software product line (SPL) approach are often fewer than expected. Among others, one important reason for this low efficiency is the non-systematic mapping of customer requirements, due to the fact that requirements engineers are often not sufficiently informed about SPL characteristics. In this research preview, we present our current and planned work regarding this problem. We discuss how we plan to develop an approach that systematically guides the tailoring of elicitation instruction for application engineering requirements engineering (AERE). In this regard, we also discuss expected improvements and how we plan to evaluate them.

1 Introduction

As a key concept for fast and efficient software development, software product lines (SPL) [1] have proven to be a promising strategy. Nevertheless, developing new systems based on an SPL during the application engineering (AE) phase is still time-consuming and expensive [2], and the benefits from using an SPL approach are often fewer than expected [3].

Among others, one important reason for this low efficiency is the non-systematic mapping of customer requirements [4], even though it has been recognized that the success of AE mainly depends on how requirements are treated. On the one hand, current application engineering requirements engineering (AERE) approaches foster the direct reuse of SPL requirements rather than the effective alignment of a customer's actual needs with the available SPL capabilities [5] [6]. However, customer-specific development based merely on picking predefined requirements from a repository is typically not sufficient, because such systems also have to reflect a significant number of individual requirements in order to allow a customer to stand out from the competition. This is especially the case in the information system (IS) domain, in which very specific business processes have to be mapped to IT. Therefore, many costly rework iterations are typically needed until a delivered system fulfills its expectations.

On the other hand, eliciting customer requirements from scratch without considering SPL characteristics early on is also not an appropriate option. Particularly since selecting an SPL implies a certain set of constraints, it becomes apparent that not all customer requirements can be realized as initially stated. Rather, trade-offs between ideal requirements and rapid development benefits must be made. However, making this trade-off is a challenging task, because information about the realizability of requirements is often neither formalized nor available in the early requirements phase. As a result, it is therefore often up to the development team in the back office *"to bridge the gap between requirements and implementations"* [7]. This typically leads to either costly rework or renegotiations until the number of non-fitting requirements is reduced. Unfortunately, checking the fit in such an analytical manner and improving the requirements afterwards is still the state of the art. Approaches, that handle this challenge better, do not exist yet [4] [5]. Thus, the question *how a satisfactorily fit between customer requirements and SPL characteristics can be achieved much faster* is still an unsolved practical problem to be addressed.

In our previous work, we have identified the fact that *AE requirements engineers are often not informed sufficiently about the underlying SPL* as a reason for this problem. We therefore currently deal with the research question on *how an AE requirements engineer can be enabled to use knowledge about a SPL for better guiding the elicitation of customer requirements.* This includes the sub questions on how to extract SPL knowledge and on how to represent this knowledge appropriately.

As a general idea to answer these questions, we have proposed the idea to tailor AERE processes based on the characteristics of a given SPL, and to provide very precise and informative elicitation instructions in this regard (see [8] [9]). Our fine-grained research objectives (RO) arising from this aim include a

(1) **clarification of the role of requirements in AE** in order to explain how AERE processes should be aligned with a given SPL,

(2) **definition of an appropriate structure for AERE elicitation instructions** that enable requirements engineers to work more effectively,

(3) **formalization of tailoring steps** in order to provide a precise procedure for a reproducible reflection of SPL knowledge in these elicitation instructions,

In this paper, we present our current work regarding the first objective (see section 2) and we give an outlook how this clarification may support our ongoing and future research regarding objectives 2 and 3 (see section 3). Thus, the main contribution of this paper is a first clarification which requirements should be in the focus of AERE and how this knowledge can be used to design an AERE tailoring approach.

2 Current Research and Results

An important prerequisite to elaborate a tailoring approach that reflect SPL knowledge in an AERE process is a clear understanding on how AERE processes are related with an underlying SPL (RO 1). Our research effort spent so far has therefore dealt with the clarification of the role and interrelationship of requirements in AE.

Basically, *"requirements express the needs and constraints placed on a product that contributes to the solution of some real world problem"* [11]. However, as there is still no universal definition of a requirement [10], several taxonomies have been proposed in the RE standard literature in order to clarify what a requirement could be. What all these taxonomies have in common is that they only address the content of a requirement (e.g., functional vs. non-functional) and not the role a requirement may have with regard to development or reuse. In many of today's AE projects, however, one can observe that the set of **elicited** customer requirements is only partially **fitting** a given SPL. Based on our own experience with RE in industry projects, we have therefore recently developed a novel taxonomy that distinguishes requirements from the viewpoint of their relevance and realizability in AE.

As fitting requirements, we basically define those elicited requirements that cover the SPL's **variable** requirements or the SPL's **implicitly anticipated** requirements (i.e., those that are in-scope but not explicitly modeled already), and those elicited requirements that are not in scope but **incidentally realizable** anyway. Fitting requirements have therefore all in common that they are (economically) **realizable** with a given SPL and also needed for its instantiation or customer-specific extension. In this regard, we call the entire and probably infinite super set of all variable, implicitly anticipated, or incidentally realizable requirements a SPL may realize, the set of **relevant** requirements for this SPL. However, another, typically not small, subset of elicited requirements is often neither needed nor desired. First, there are requirements that are **unnecessary** because the solutions they ask for are implemented by default anyway. Hence, these requirements just repeat **common** SPL requirements. Second, there are elicited requirements that are not (economically) realizable with the given SPL. These **problematic** requirements could only be fulfilled with very high costs and risks because significant changes have to be made either to common implementations or even to the entire architecture. Problematic requirements typically result from an insufficient consideration of existing constraints and thus, from giving customers too much freedom. Besides problematic requirements, **missing** requirements are another risk. In particular, when developers do not get all the information they need for making decisions, they have to make their own assumptions or re-elicitations. Missing requirements typically arise from the fact that no concrete instruction is given on how to elicit which requirements.

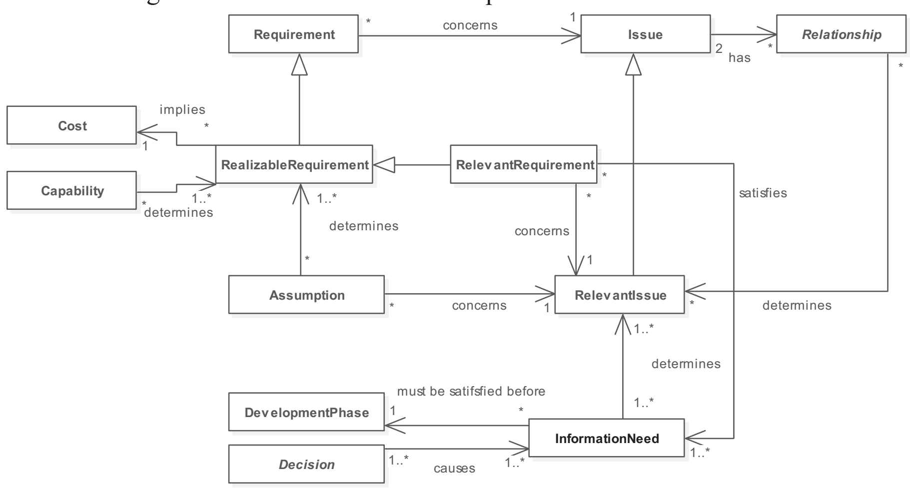


Fig. 1. The origin and relationships of relevant requirements

In order to better align AERE processes in a sense that all requirements are fitting and no important requirement is missing, we have currently developed a conceptual model with around 75 elements in ten views that clarifies how AERE processes should be aligned with a given SPL. In this model, from which we only show two views in this paper (see Fig. 1 and Fig. 2), especially “information needs” and “assumptions” are essential elements to explain how knowledge about a SPL can be appropriately extracted.

Basically, a **requirement** is a piece of information concerning an issue. As **issue** we define a class of elements that are either part of a system or part of the system’s usage environment. Issues therefore cover all functional and non-functional system aspects (e.g., use cases), but also elements of the usage environment for which a system must be designed in order to provide appropriate support (e.g., users, work places, data, devices, etc.). As issues represent the real world, there may be **relationships** between them; for instance, between users and their workplaces.

However, as mentioned before, not all requirements are (economically) realizable with a given SPL. The important drivers for determining **realizable requirements** are the already existing, explicitly anticipated requirements, as well as the set of **assumptions** that have been made during the design of the product line architecture. Thus, assumptions are descriptive rather than enumerative specifications of requirements that are realizable by a SPL. For requirements concerning the issue “user interface”, for instance, an assumption could be that only web-based clients are (economically) realizable. Of course, when considering the SPL’s common requirements, it becomes evident that not all **realizable requirements** are also relevant for deriving systems from an SPL. To refine the aforementioned definition, a relevant requirement is thus a realizable requirement that is concerned with a **relevant issue**. In this context, an issue is called relevant if a requirement concerned with it is able to satisfy an **information need** in the AE’s development process, or, if an issue is needed for the elaboration of such requirements. For instance, when information about workplaces is necessary to implement an appropriate user interface, “workplace” would be a relevant issue. And, also the “user” via whom the “workplaces” are to be identified, would be relevant implicitly.

Thus, besides assumptions, information needs are another central concept in our conceptual model to determine which requirements are relevant and which are not. In order to identify these information needs, the **decisions** to be made during an AE’s development process must be analyzed, as each decision basically causes at least one information need. In the workplace example, the decisions on how to implement a user interface have caused the need to know what the workplaces look like. Thus, before a **development phase** dealing with the development of user interfaces can start, this information need must be satisfied through the provision of realizable requirements concerning the workplaces to be supported.

3 Open Issues and Outlook

It is evident that knowledge about both the development processes and the product line architecture must exist in order to determine relevant requirements through the notion of information needs and assumptions. Currently, we are finalizing our conceptual model that explains how all SPL-related aspects belong together. An open issue in this regard is whether our model is correct and sufficiently complete. To answer this question, we plan to validate the model through expert reviews soon.

Besides the conceptual model, we have recently started to develop a tailoring approach (addresses RO 3) that explains how assumptions and information needs can be systematically extracted from a SPL and then reflected into precise elicitation instructions. The purpose of these elicitation instructions is to prescribe the entire AERE process including all activities that are necessary for the elaboration of realizable requirements concerning all relevant issues (see Fig. 2). In each requirements activity, the elicitation instruction must therefore inform about the capabilities and assumptions of the SPL. By using this information, a requirements engineer is then able to guide the elicitation better, and to initiate negotiation as soon as the expectations of a customer seem to contravene the given SPL constraints.

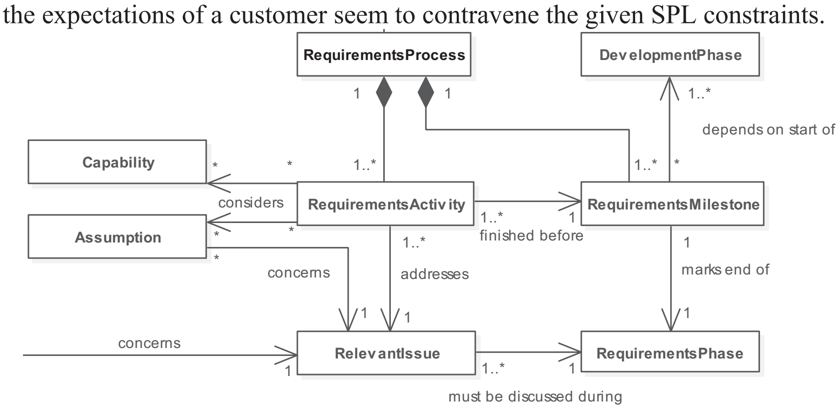


Fig. 2. Structure and relationship of an AE requirements process

In the aforementioned example, the need to know something about workplaces would lead to a corresponding requirements activity in which a requirements engineer has to elicit information about the workplaces at which the system should be used. During this step, assumptions that the SPL makes regarding the nature of workplaces (e.g., visualization power of the devices) is used to check early on whether an appropriate user interface could be developed easily. If not, the requirements engineer is immediately able to start negotiations, or to inform the customer about extra costs, even if concrete requirements have not been anticipated.

However, an open research issue is the right degree of detail the tailoring approach must have. This is actually a problem as some tailoring steps seem to be easily describable in an almost algorithmic manner, while others are hard to prescribe at all.

Furthermore, not only the tailoring approach, but also the resulting elicitation instructions need an appropriate degree of preciseness. In order to reflect the assumptions and relevant issues in a form that is suitable for AE requirements engineers, we have therefore recently performed a survey with RE experts to elicit requirements these elicitation instructions should fulfill (addresses RO 2). Now, we are faced with the challenge on how to implement these requirements in an elicitation instruction template including the definition of section structures, text modules, or content-generating algorithms. This is also a prerequisite to develop a supporting tool.

Regarding evaluation, we have already performed a feasibility study very early in our research in order to show that AERE tailoring is basically possible, and that the effort for performing this tailoring is justifiable (see [9]). Furthermore, we have calculated possible benefits when using our elicitation instruction in comparison to traditional AERE artifacts through a simulation on how different approaches deal with the mentioned types of elicited requirements. According to this simulation, we expect a possible saving of up to 38% rework effort when using instructions according to our approach. Of course, an open question is whether these improvements are actually achievable in reality. Thus, we still have to confirm these calculations through empirical studies. From a scientific point of view, we plan to evaluate in a controlled experiment that elicitation instructions based on our tailoring approach provide requirements engineers better (more complete and / or faster) with information about a SPL than traditional AERE artifacts do. From a rather practical point of view, we plan to conduct case studies in industry for demonstrating that the use of such (better informing) instructions can also lead to a faster achievement of a satisfactory fit between requirements and SPL characteristics.

4 Conclusion

In this paper, we have presented our current and planned work regarding the problem that a satisfactorily fit between customer requirements and SPL characteristics cannot be achieved sufficiently fast during AE. To cope with this problem, our research aims at providing an AE requirements engineer with better SPL knowledge by means of tailored elicitation instructions. While basic work such as the performance of a feasibility study or the definition of a conceptual model is (almost) done, the development of the actual tailoring approach, the definition of an elicitation instruction template, as well as the performance of a final evaluation is still open.

References

1. Clements, P., Northrop, L.: Software Product Lines. Addison Wesley, 2001
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. Rabiser, R., Grünbacher, P., Dhungana, D.: Supporting Product Derivation by Adapting and Augmenting Variability Models. In: SPLC. IEEE, 2007
4. O'Leary, P., Rabiser, R., Richardson, I., Thiel, S.: Important Issues and Key Activities in Product Derivation: Experiences from Two Independent Research Projects. In: SPLC 2009
5. Djebbi, O., Salinesi, C.: RED-PL, a Method for Deriving Product Requirements from a PL Requirements Model. In: CAiSE. Springer, 2007
6. Guelfi, N., Perrouin, G.: A Flexible Requirements Analysis Approach for Software Product Lines. In: Proceedings of RefSQ 2007, LNCS 4542. Springer, 2007
7. Baum, L., Becker, M., Geyer, L., Molter, G.: Mapping Requirements to Reusable Components using Design Spaces. In: Requirements Engineering Conference. IEEE, 2000
8. Adam, S.: Improving SPL-based Information System Development Through Tailored Requirements Processes. In: Doctoral Symposium @ RE 2010
9. Adam, S., Doerr, J., Ehresmann, M., Wenzel, P.: Incorporating SPL Knowledge into a Requirements Process for Information Systems. In: PLREQ @ REfSQ 2010. Essen, 2010
10. Wiegers, K.: Software Requirements – Second Edition. Microsoft Press, 2005
11. IEEE's Software Engineering Body of Knowledge. IEEE, 2004

Pragmatic Variability Management in Requirement Specifications with IBM Rational DOORS

Ekaterina Boutkova

Group Research and Advanced Engineering
Daimler AG
Wilhelm-Runge-Straße 11
89081 Ulm, Germany
ekaterina.boutkova@daimler.com

Abstract. [Context and motivation] Reuse of requirements leads to reduction in time spent for specification of new products. Variability management of requirement documents is an essential prerequisite in terms of a successful reuse of requirements. It supports the decisions whether existing requirements can be reused or not. One possibility to document the variability is feature modelling. **[Question/problem]** A problem for the application of the feature modelling approach in requirements specification is a lack of tool support. An analysis of the existing tools shows some deficiencies that render these tools unsuitable for the needs of Daimler passenger car development (Daimler PCD). **[Principal ideas/ results]** An extension of requirements management tool IBM Rational DOORS functions can help to solve the challenge of tool support in many typical cases at Daimler PCD. **[Contribution]** This paper presents a landscape of the requirement specifications at Daimler PCD and a survey of old and new approaches of variability management. Furthermore, the requirements on tool support are discussed and a pragmatic extension of the requirement management tool DOORS are presented.

Keywords: DOORS, feature modelling, requirement, specification, variability, variability management

1 Introduction

Specifications play an important role in the development process of high tech products. They are all-embracing and the specification process itself is very costly in terms of time and effort. The reuse of requirements could help to reduce the time spent for development with regard to variability diversity [3]. Variability management (VM) of requirements is an essential prerequisite in terms of a successful reuse of requirements. A basic approach for requirements reuse in practice, at least at Daimler passenger car development (Daimler PCD), is to tag each requirement for each car model the requirement is valid for. Given the fact that specifications for a single component usually consist of several hundred to several ten-thousand requirements [5] it is obvious that such an approach is labor-intensive and error-prone. One promising possibility to document the variability in specifications is feature modelling [6]. It bases on the assumption that there are product properties (features) that are the drivers for similarities or dissimilarities.

There are not too many tools on the market which support the feature modelling approach and are compatible with DOORS [4]. At the same time DOORS is the standard tool for requirements management in the industry (e.g. automotive domain). DOORS itself has no functions for requirements management for product lines. The analysis of the tools on the market shows that none of these tools fulfils all the defined requirements from Daimler PDC. Thus the decision was made to build DOORS extensions that provide the necessary functionalities to implement a feature driven VM.

This paper presents a lean approach for VM in requirement specifications. This approach builds on feature modelling [6] and is tailored for use with the requirement management tool DOORS both for variability documentation and variability management. The new approach allows creating of a new specification for a product variant within 15-30 minutes. Therefore the novel approach radically reduces the time effort for creating variant specifications.

The next section briefly describes the requirement specifications landscape at Daimler PCD. Section 3 describes approaches for variability management in requirement specifications that have been used so far in most cases in Daimler PCD specifications. Along with the description of the specification landscape in Section 2, this provides a good description of the situation in which an enhanced but lean variability management approach has to work in. Additionally, we explicitly identify some challenges and boundary constraints an enhanced variability management approach has to cope with. Section 4 discusses why available tools were not adequate in our situation and presents our solution for lean variability management within DOORS. An example illustrates the current solution along with its impacts on the requirements engineering work practice. Section 5 presents first experience of the using the DOORS extensions and the feature based variability management (FBVM). Section 6 provides a short conclusion.

2 Requirement Specifications

At present, three abstraction levels of specifications exist at Daimler PCD. These levels are *vehicle specifications*, *system specifications* and *component specifications*.

A *vehicle specification* documents the marketing view (e.g. target customer group, product positioning on the market) as well as the design and technical concepts for the new car as a whole.

A system (as the term is used at Daimler PCD) consists of tangible customer functions and characteristics of these. Typical examples of systems are Outside Light Control or Seat Heating. A system is typically implemented as interplay of many components. Outside Light Control, for example, is implemented by up to 22 ECUs [8]. Thus, a *system specification* builds a bridge between vehicle requirements and component requirements as it decomposes the vehicle-level requirements on the systems and allocates detailed requirements to the contributing components.

A *component specification* is a central technical specification and is basis for tendering and contract with a supplier.

Around 100 system specifications and about 400 component specifications [5] will be created for one new car model. The classification of Regnell et al. [11] places many of these specifications into the category “large-scale RE” (order of 1,000 requirements). Some specifications contain up to 50,000 individual elements (DOORS objects) and therefore fall in the category "very large-scale RE" [7]. Most system and component specifications are written by using DOORS.

3 Variability Management in Requirement Specifications

This section presents the overview of the existing approaches for the VM at Daimler PCD and describes challenges as well as requirements for VM approaches from Daimler PCD. At last, an enhanced approach, which based on feature modelling, will be presented.

3.1 State of the Practice

All approaches presented in this section as well as the whole vehicle development at Daimler PCD are structured through the car models. If a new car model is planned, all existing requirement specifications will be analysed if the reuse of requirements is possible for the needs of the new car series. The analysis of all existing approaches for the variability management in specifications is presented in [1]. This paper gives only a short overview.

ID	REQUIREMENT
1	The light must be activated through the turn down of sun visor [E-Class, S-Class]
2	The décor material of the sun visor must be drapery.
3	The décor material of the sun visor must be leather [E-Class , S-Class]

Fig.1. Attribute Columns Approach.

The documentation of the variants in *the requirement text* is a first possibility. In this case, the variants are a part of a requirement. Figure 1 shows an example. This notation means: The requirement (1) is valid for E- and S-Class but not for A-Class. The requirement (2) is valid for A-, E- and S-Class. The requirement (3) is valid for E- and S-Class but not for A-Class

ID	REQUIREMENT	A	E	S
1	The light must be activated through the turn down of sun visor.	-	YES	YES
2	The décor material of the sun visor must be drapery.	YES	YES	YES
3	The décor material of the sun visor must be leather.	-	YES	YES

Fig.2. Attribute Columns Approach.

The *attribute columns* approach uses attribute columns in DOORS in order to document the variants. Each variant receives one column including the following possible values, "yes" (the requirement is valid for this variant) or "-" (the requirement is not valid for this variant). Figure 2 shows the example as above. Most of the time, the columns represented new car models. Sometimes, the columns represent the components (e.g. front and rear bumper). Additionally, there were cases where the columns represent a combination of a car model and a component or car body type.

ID	REQUIREMENT	CAR MODEL
1	The light must be activated through the turn down of sun visor.	E-Class S-Class
2	The décor material of the sun visor must be drapery.	common
3	The décor material of the sun visor must be leather.	E-Class S-Class

Fig.3. Multi-Enumeration Value Attribute.

The approach *multi-enumeration value attributes* is a modification of the attribute columns approach. Car series values will be documented in one column with multi-enumeration value attribute. Additionally, there exists the "common"-value, which marks requirements that are valid for all variants. Figure 3 shows the same example as Figure 2.

The *chapter approach* is based on the differentiation between common and the variant-specific requirements. The common (or variants-spanning) requirements are documented in an appointed part of the specification. This part is placed in each specification. If the variant requirements differ from overlapping requirements, they will be documented in the variant specific part.

The analysis of current approaches shows some shortcomings. The main advantage of these approaches are (1) bad scalability, (2) high effort for each new specification document (as all requirements has to be classified in the context of the new specification document), and (3) no support in modelling dependencies between requirements.

3.2 Challenges and Requirements

Hubaux et al. [14] analysed the challenges on feature modelling faced by practitioners. All of the identified challenges on VM [4] match the challenges, which were identified at Daimler PCD [2]. This section exemplifies most important challenges and boundary conditions (i) to (iv).

(i) *Size and Complexity of Variability Models and Specifications*. There are many reasons for variability in one car. One simple component has 10-100 possible variation points. So, one car has some thousands of variation points which must be documented and managed. Another facet is the size and heterogeneity of specifications. Most requirement specifications at Daimler PCD are very large. However, small requirement specifications (e.g. for a standard part as a bolt) also exist. So, the range in specification size is very large. The range between the number of variable requirements as well as the range between the numbers of component or system variants is also very large.

(ii) *Human Factor*: The component and system specifications at Daimler are written by engineers. They are trained experts in the areas of the vehicle, system and component development. They have corresponding education as mechanical or electrical engineers. That is why most of them do not have in-depth know-how in information technology in general and requirement management in specific. Furthermore the time for training is limited. The main task of the engineers is the component or system development. In a typical vehicle project, they spend two to six months on writing their specification document. The next three to five years, the engineers "only" accompanies the development of his component, a job which consists of many tasks – most of them are not related to requirements management.

(iii) *Requirements Management Tool*: The requirements documents at Daimler PCD are documented using DOORS.

(iv) *Specification Language*: A further challenge is that specifications at Daimler PCD are mainly documented in natural language.

During the analysis of the existing approaches for VM in specifications and many interviews with experts at Daimler PCD, the following requirements for a variability management approach were identified:

(R1) There has to be no redundant documentation of the same requirement in the specification or requirement library.

(R2) The requirements that are valid for all variants must be recognizable.

(R3) The mapping between requirement and variant must be understandable for everyone (i.e. especially for non-variability-management-experts).

(R4) It must be possible to compare the different variants in the specification directly.

(R5) The mapping between requirements and product variants must be semi-automatically testable. For example, if there are two requirements that are mutual exclusively, there must be an effective way to check this.

(R6) The variability management approach must be scalable, i.e. it must be applicable to specifications that consist of several ten-thousands of requirements.

The next section describes the Feature Based Variability Management (FBVM) approach.

3.3 Feature Based Variability Management

The result of the analysis of current approaches at Daimler PCD shows that an improved approach for variability management in specifications is needed. The literature research shows that the orthogonal [9] feature modelling [6] has a high potential for variability documentation of the natural language specifications. Feature modelling is already applied at Daimler PCD in some projects for variability management during design and implementation phases. The adoption of feature modelling for variability in specifications could help to have a continuous variability management at all development phases.

The basic principle of feature modelling is to show not only the variable characteristics of a product (optional or alternative features) but also the common characteristics of all products from the same product line (mandatory features).

The *feature modelling* was introduced by Kang et al. [6] as a part of Feature Oriented Domain Analysis (FODA). The main idea of the feature modelling is to create an abstraction level above requirements in order to support visualization, documentation, and the communication between stakeholders of product variants. FODA defines features as *"user-visible aspects or characteristics of the domain"*[6]. In this paper, feature is defined as *a characteristic of one product of the product line, which it could have*, i.e. a feature does not necessarily have to be user-visible. Architectural properties, for instance, could also be a feature. This modelling technique was refined in many researches and is highly accepted in industry [8]. There are many variations of feature modelling approaches, which were developed to solve different problems in various domains [8]. The exact description of the feature modelling technique is not the aim of this paper and could be taken from [6, 8].

During the analysis of the existing approaches for the VM in specifications, three levels of need were identified at Daimler PCD.

- At the first level, only the variable features (optional or alternative features) must be documented. There are no relationships between features. The number of features is low (about two to seven parent features, each with two to five leaf features).
- At the second level, there are some relationships between features. The number of features is middle (about two to 15 parent features, each with two to ten leaf features).
- At the third level, all features (common and variable) must be documented. There are many relationships between features. The number of features is high (about 30 to 50 parent features, each with ten to 30 leaf features).

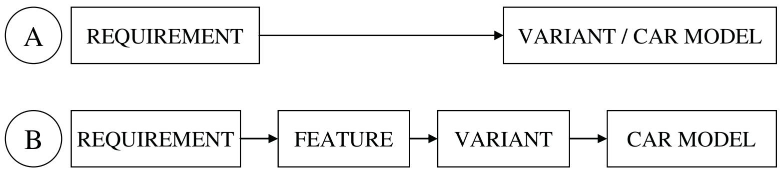


Fig.4. Mapping of Requirements.

An important precondition for the application of FBVM in specifications is the necessity of a paradigm change. The idea is the separation of requirements and car models (Fig. 4A). In case of feature modelling, the requirements should be mapped to features (Fig. 4B). This mapping from requirement to feature creates a non-recurring effort, because in most cases it does not change anymore. A new variant (e.g. a new car model) is specified by including or excluding of features from the variant model.

FBVM approach was developed considered to the needs of all development departments (e.g. mechanics, eclectics and electronics). The different values of the methodical needs (e.g. the electronic components have mach more features and relationships between features as mechanic components) were defined and the corresponding stages of the methodical support were work out. FVBM approach fulfils all requirements R1 to R6 on the VM approach from Section 3.2.

4 Feature Based Variability Management with DOORS

This section describes at first the shortcomings of the tool support in the area of variability management. Then the requirements on tool properties will be presented. Section 4.2 presents the extensions of DOORS functions, which enable to use DOORS for variability management in specifications. At last, the application of the enhanced variability management approach – which is supported through DOORS extensions – is presented.

4.1 Feature Modelling Tools

There are not too many tools on the market which support the feature modelling approach. DOORS is the base requirement tool at Daimler, so it is understandable that a first requirement for VM tool is compatibility with DOORS. There are three tools on the market which fulfil this requirement: Gears from BigLever [12], pure::variants [14] and metadoc [13]. The further requirements for tools are:

(TR1) The mapping between requirement and variant must be easily understandable. This means that no code or Boolean operator can be used for this mapping.

(TR2) The structure of a specification may not be changed by a VM tool.

(TR3) The comparison of different variants in the same requirement specification must be possible inside this specification.

(TR4) The variability model may not be changed from the specification.

At last, a wish of most users was to have the possibility to document and manage variability in the same tool as the requirements are documented in, i.e. inside DOORS.

None of the above mentioned tools fulfils all requirements (TR1-TR4). Especially for the first two levels of complexity (see Section 3.3), intense trainings and the introduction of a new tool or new specification structures seemed to be not adequate. Thus we decided to build some DOORS extensions that provided the necessary functionalities to implement a feature driven variability management.

For more complex situations (which could be characterized as complexity level three), we decided to invest in trainings and to use a tool that is capable to deal with complex feature models. Here, we selected pure::variants.

4.2 Variant Module in DOORS

The v*ariant module* is an extension of DOORS functionality. It provides the possibility to document and manage the variability within DOORS. These extensions were programmed in the DXL and Java programming language [15]. The base of the variant module is one formal module and two link modules in DOORS. The formal module documents the variability. The first link module contains links inside of formal module. The impact of the link is dependent on the object type (AND, OR, NOT, IF-THEN). The second link module supports the mapping between the variant module and specification modules.

Table 1. Object Type Values

Attribute	Description
Source-ID	This attribute gives the unique number for each object in the variant module
Object Type	This attribute makes it possible to document the features and the relationships between these features. The possible values of this attribute are: - A *criterion* denotes the parent features. - A *criteria value* denotes the leaf features. Each feature has at least two leaf feature (exist and not exist). - A *combination* is the selection of criteria values considering the rules. The combination can be created through links, which connect (AND) the required criteria values. - A *configuration* is the selection of criteria values and/or combinations considering the rules. The configuration can be created through links, which connect (AND) the require criteria values and /or combinations. - A *negation* is an auxiliary construct for the logical operator NOT. The out-link from a negation object shows on the negated criteria value. - A *group* is an auxiliary construct for the logical operator OR. - A *rule* describes the relationship between two criteria values. The in-link from rule denotes the condition (IF) and the out-link from rule denotes the effect (THEN).
Object Text	This attribute makes it possible to document the names of features, combinations, configurations and rules.
Object Short Text	This attribute makes it possible to document the user abbreviation for names of features, combinations, configurations and rules.
Visible	This attribute has binary states {false, true} and makes it possible to manage which features and which configurations will be available in the specification.
Simple Formula	This attribute shows the formal composition of combination, configuration and rules.
Variant Document	Variant Document denotes the first object of variant module. This is the precondition for the mapping with specification modules.

The variant module has six columns (which are the representation of the corresponding attributes) for the variability documentation. These attributes are presented in Table 1. The variant module template consists of four parts, which are: features, combinations, configurations, and rules. In the "feature part" all features can be documented. In the "combination part" valid combinations of features can be documented. In the "configuration part" valid combination of features, which are the required variants of components or systems, can be documented. The difference between combination and configuration is that a combination could be a part of a configuration. Therefore, a configuration can consist of features and combinations.

Additionally, the variant module supports the mapping from requirements to features and provides the following functions:

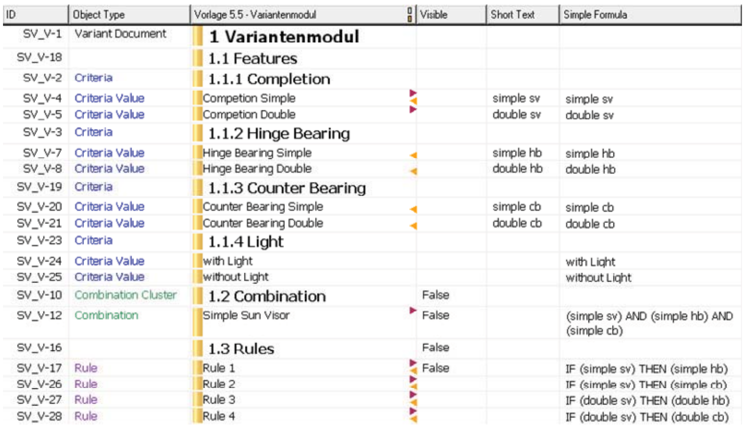
- Propagate the feature mapping from parent requirement to a child requirement.
- Propagate the feature mapping from child requirement to parent requirement.
- Propagate the feature mapping from parent requirement to all children requirements.

4.3 FBVM with Variant Module in DOORS

The application of the FBVM approach in DOORS will be described with the sun visor specification as example. The sun visor was chosen for this case study because its specification is not too large. There are about 400 technical requirements. In the initial specification all requirements are mapped to the next three car models.

The first step is to identify both the common and the variable requirements. Afterwards, it is possible to identify the features, which are the rationales for the requirements variability. Currently this process bases on manual reviews.

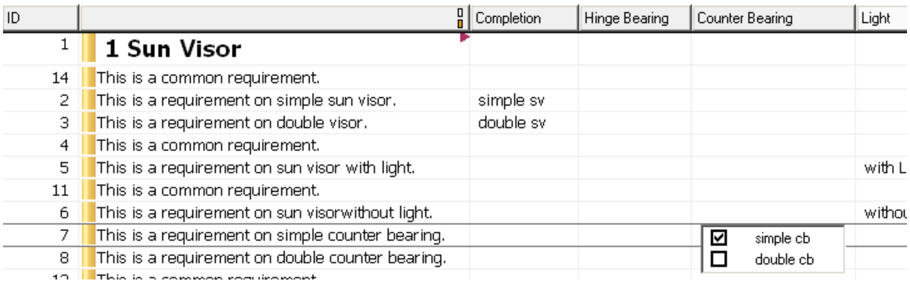
The second step is the feature documentation in the variant module. The parent features (e.g. completion, hinge bearing) have the *Object Type Value "Criteria"*. The leaf features (e.g. simple, double) have the *Object Type Value "Criteria Value"*. The hierarchic relationships between parent features and leaf features are documented with the standard DOORS object hierarchy. The "needs" relationships between features are documented with rules. Fig. 5 shows the variant module for the component sun visor.



ID	Object Type	Vorlage 5.5 - Variantenmodul	Visible	Short Text	Simple Formula
SV_V-1	Variant Document	1 Variantenmodul			
SV_V-18		1.1 Features			
SV_V-2	Criteria	1.1.1 Completion			
SV_V-4	Criteria Value	Competion Simple		simple sv	simple sv
SV_V-5	Criteria Value	Competion Double		double sv	double sv
SV_V-3	Criteria	1.1.2 Hinge Bearing			
SV_V-7	Criteria Value	Hinge Bearing Simple		simple hb	simple hb
SV_V-8	Criteria Value	Hinge Bearing Double		double hb	double hb
SV_V-19	Criteria	1.1.3 Counter Bearing			
SV_V-20	Criteria Value	Counter Bearing Simple		simple cb	simple cb
SV_V-21	Criteria Value	Counter Bearing Double		double cb	double cb
SV_V-23	Criteria	1.1.4 Light			
SV_V-24	Criteria Value	with Light			with Light
SV_V-25	Criteria Value	without Light			without Light
SV_V-10	Combination Cluster	1.2 Combination	False		
SV_V-12	Combination	Simple Sun Visor	False		(simple sv) AND (simple hb) AND (simple cb)
SV_V-16		1.3 Rules	False		
SV_V-17	Rule	Rule 1	False		IF (simple sv) THEN (simple hb)
SV_V-26	Rule	Rule 2			IF (simple sv) THEN (simple cb)
SV_V-27	Rule	Rule 3			IF (double sv) THEN (double hb)
SV_V-28	Rule	Rule 4			IF (double sv) THEN (double cb)

Fig.5. Sun Visor Variant Module in DOORS

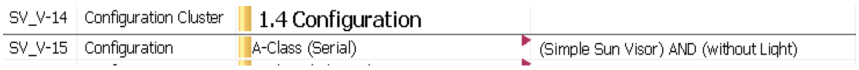
The next step is the mapping of requirements to features. At first, the specification module must be mapped to the variant module. Then, features from the variant module must be imported into the specification. The features will be imported as attribute with multi-enumeration values (equal to leaf features names). For each feature, a column will be created. The requirements that are valid for all features do not have any mapping to features. The variable requirements must be mapped to correspondent features. Fig. 6 shows the mapping in the sun visor specification.



ID		Completion	Hinge Bearing	Counter Bearing	Light
1	1 Sun Visor				
14	This is a common requirement.				
2	This is a requirement on simple sun visor.	simple sv			
3	This is a requirement on double visor.	double sv			
4	This is a common requirement.				
5	This is a requirement on sun visor with light.				with L
11	This is a common requirement.				
6	This is a requirement on sun visorwithout light.				withou
7	This is a requirement on simple counter bearing.			☑ simple cb	
8	This is a requirement on double counter bearing.			☐ double cb	

Fig.6. Specification Module

If a new variant of the sun visor is required, a new configuration must be created in the variant module. Fig. 7 presents the new variant of the sun visor. It has simple completion, simple hinge bearing, simple count bearing and no light. It is now possible to create the variant specific sun visor specification in only a few minutes. In the specification module, the filter function must be activated. The result of this filtering can be saved as variant view which represents the wanted specification.



SV_V-14	Configuration Cluster	1.4 Configuration	
SV_V-15	Configuration	A-Class (Serial)	(Simple Sun Visor) AND (without Light)

Fig.7. New Configuration in Variant Module in DOORS

5 Discussions

This section presents the first the experiences with the application of the FBVM in specifications at Daimler PCD as well as the experience with the variant module in DOORS.

An important step during the introduction of the FBVM was the definition of all relevant terms (e.g. *feature*, *variant, and product*). It was necessary because a common language is the precondition for the successful collective work of many people with different education and practice experiences.

The development of the migration process from *attribute based approaches* to the FBVM approach was a challenge. This transition is currently done manually. Especially costly in terms of time is the manual identification of the features. This process takes about one day for an average component specification document. Therefore, a semi-automatic feature identification approach was developed and is currently being evaluated.

In order to manage the complexity of vehicle variability, a solution with *decentralized variability models* was chosen. In the *decentralized variability models* approach a variability model for each component and each system must be created. The decentralized variability models allow to reuse the requirements and configurations for individual components and systems in a short time. The support of the configuration process at the vehicle level as well as the synchronisation of variability models at all levels are the next challenges.

Particular attention was spent on the acceptance of the new VM approaches from engineers and support group. In order to win the acceptance, multiple presentations of the FBVM approach were offered. Furthermore, the pilot applications were supported not only through the support group but also through the developer of the FBVM experts. This course of action allowed collecting experience with this new approach and to find the weaknesses and possible improvements.

The first experiences with the FBVM are positive. FBVM is well accepted by the pilot users. The pilot users at the component level develop mechanical components with few features (not more than ten features) as well as electronic components with about 70 features and 10^{16} theoretical possible variants. At the system level, the pilot users develop systems with about ten features. The main advantages of FBVM are:

(1) The reduction of the time effort for creating the variant specifications. The creation of a specification for the new car model with the *attribute column approach* for a specification with 1.000 requirements takes about 8 hour (it means 30 second for analysing and mapping per requirement). This mapping must be repeated for each new car model. The introduction of the FBVM for a specification takes about 8 hour for the identification of features and the building of the variability model. This effort is nonrecurring in contrast to the mapping of requirements to car models. The creation of a specification for the new car model with FBVM for a specification with 1.000 requirements takes about 15 minutes.

(2) The know-how about reasons for the variability. The features show the rational for the existence of variability. During the application of the *attribute column approach* this information was hidden.

(3) Independence from a specific requirements management tool.

One disadvantage of FBVM approach is the effort for the training of the engineers and support groups. The training consists of the presentation and the building of the variant module and it takes about five hours. The further disadvantages are the initial effort for the creation of the variability models and the necessary adjustments of the processes following the specification phase.

The first experiences with DOORS extensions are positive, too. These extensions allow documenting and managing the variability within DOORS. Therefore, an engineer doesn't need to use different tools. Most of the variant modules are not too large and consist of not more than ten parent features. Many analysed component and system variability models at Daimler PCD match this condition. In other cases, the same variability management approach can be implemented using the tool pure::variants [14].

6 Conclusion

This paper presents pragmatic extensions of the requirement management tool DOORS. These extensions support a new FBVM approach for the documentation and management of variability in natural language specifications. The extensions of DOORS functions make it possible to document and manage the variability in requirements with DOORS.

The presented DOORS extensions are suitable for smaller numbers of features with no complex relationships between them. The typical well supported number of features is about ten parent feature and five leaf features. Many analysed component and system variability models at Daimler PCD match this condition.

The FBVM approach and extensions of DOORS functions do not solve all challenges on VM in specifications. But this approach including the presented DOORS extensions make it possible to effectively and efficiently reuse requirements in specifications at Daimler PCD right now. New specifications for new product variants can be configured from the corresponding feature models within a few minutes.

There are some related works, which try to manage the variability in specifications or in technical documentations. Nicolás et al [10] gives a good overview about existing approaches for VM in specifications. The main difference to all another approaches is that the presented approach allows keeping the existing specifications language, the existing structure of specifications as well as the requirements management tool.

References

1. Boutkova, E.: Variantenmanagement in Anforderungsdokumenten: State of the Practice (Variability Management in Requirement Specifications: State of the Practice). In: Softwaretechnik -Trends 9(1) (2009) in German
2. Boutkova, E.: Herausforderungen für Variabilitätsmanagement in Anforderungsdokumenten (The Challenges for Variability Management in Requirement Documents). PIK 2010, Paderborn (2010) in German
3. Cheng, B.H.C., Atlee, J.M.: Research Directions in Requirements Engineering. In: Future of Software Engineering (FOSE'2007), IEEE (2007)
4. Hubaux, A., Classen, A., Mendonca, M., Heymanns, P.: A preliminary review on the application of feature diagrams in practice. In: Proceeding of VaMoS 2010, 53-59 (2010)
5. Houdek, F.: Challenges in Automotive Requirements Engineering. Industrial Presentations by REFSQ 2010, Essen (2010)
6. Kang, K.C., Cohen, G., Hess, S.J.A., Novak, W.E., Spencer, A.: Feature-Oriented Domain Analysis (FODA) Feasibility Study. Peterson (1990)
7. Leuser, J., Ott, D.: Tracking Semi-automatic traceability in large specifications. In: R. Wieringa and A.Pesson (Hrsg.): REFSQ 2010, LNCS 6182, 203-217 (2010)
8. Lichter, H.; von der Maßen, T.; Nyßen, A.; Weiler, T.: Vergleich von Ansätzen zu Feature Modellierung bei der Softwareproduktlinienentwicklung (The Comparison of Feature Modelling Approaches in the Software Product Line Development), (2003), in German.
9. Pohl, K.: Requirements Engineering, dpunkt Verlag, Heidelberg, 2007.
10. Nicolás, J., Toval, A.: On the generation of requirements specifications from software engineering models: A systematic literature review. Information and Software Technology, 51(9), Elsevier, 1291–1307 (2009)
11. Regnell, B., Svensson, R.B., Wnuk, K.: Can we Beat the Complexity of very Large-Scale Requirements Engineering? In: Paech, B., Rolland, C. (eds.) Requirements Engineering: Foundation for Software Quality. LNCS, vol. 5025, 123 (2008)
12. BigLever Gears http://www.biglever.com/
13. Metadoc http://www.metadoc.de/anforderungsmanagement/werkzeuge/fm/73-feature-modeller
14. Pure::Variants http://www.pure-systems.com/pure_variants.49.0.html
15. SpryLab http://www.sprylab.com/

Prioritizing Requirements: An Experiment to Test the Perceived Reliability, Usability and Time Consumption of Bubblesort and the Analytical Hierarchy Process

Geurt Johan van Tuijl, Wouter Leenen, Zhengru Shen,
Inge van de Weerd, Sjaak Brinkkemper

Department of Information and Computing Sciences, Utrecht University,
P.O. Box 80.089, 3508 TB Utrecht, The Netherlands
{g.j.vantuijl1, w.a.leenen, z.shen}@students.uu.nl
{i.vandeweerd, s.brinkkemper}@cs.uu.nl

Abstract. Software vendors often face the difficult task to deal with large amounts of requirements that enter the company every day. When dealing with this vast amount of requirements, the notion of deciding which requirements will be addressed first, and which will be addressed later, is an important decision. To support software development teams in decision-making, different prioritization techniques are discussed in previous literature. In this paper, two existing prioritization techniques called Analytical Hierarchy Processing (AHP) and Bubblesort are investigated in order to measure their outcome in terms of usability, time consumption and perceived reliability. By conducting an experiment among Dutch Master students, we discovered that Bubblesort outpaced AHP on all aspects, although this could not be supported statistically. However, based on our findings, we can conclude that there is a strong indication that Bubblesort is considered favorable compared to AHP even though it receives less attention in current literature.

Keywords: Requirements prioritization, Bubblesort, Analytic Hierarchy Processing, Software Product Management

1 Introduction

Software vendors have to handle the requirements that enter the company through customers, sales & marketing, and research & development every day. Typically, customer demand exceeds the pace at which software is developed [1], usually causing the final release to be outdated before it has been published. Eliciting the required information from customers can be difficult to achieve, especially when multiple customers with diverse expectations are involved [2]. Customer involvement is therefore a major contributing factor to company success [3].

An important issue when dealing with this vast amount of requirements is the notion of deciding which requirements are being addressed first and which will be addressed later. This difficulty, i.e. the prioritization of requirements, is recognized as an import activity in product development [4, 5] and product management [6, 7]. The process for prioritizing software requirements faces the difficult task to be simple and fast, however, also needs to provide accurate and trustworthy results [8]. A solution to this problem can be found in existing techniques such as the Binary Priority List [1], Analytical Hierarchy Process [9] and Hierarchical Cumulative Voting [10]. Regardless of which technique is carried out by the user, each technique should be able to support the product manager in decision-making but also needs to decide which requirements are the least important.

1.1 An Overview of Requirements Prioritization Techniques

Researchers, as well as professionals in industry, have proposed many requirements prioritization methods and techniques. Different techniques have been discussed considerably and compared with others in experimental studies [11], case studies [12], empirical studies [5, 13] and literature studies [18]. According to a review proposed by Racheva et al. [14], these techniques could be classified into two main categories: techniques that are applied to small amounts of requirements (small-scale) and techniques that scale up very well (medium-scale or large-scale). Examples of small-scale techniques include round-the-group prioritization, multi-voting system, pair-wise analysis, weighted criteria analysis, and the Quality Function Deployment approach. Among medium-scale or large-scale techniques, the MoSCoW technique, the Binary Priority List, the Planning Game and the Wiegers's matrix approach are frequently used. Considering the sometimes complex algorithms used in existing techniques, often supporting tools are needed to deal with large amounts of requirements.

Another classification for requirements prioritization techniques is given by Berander et al. [15]. Similar to Racheva et al. [14], they also divided existing techniques into two main categories: (1) techniques which assume that values can be assigned, by an expert, to different aspects of requirements and (2) methods that include negotiation approaches in which requirements priorities result from an agreement among subjective evaluation by different stakeholders. Examples of techniques that apply to the first category are Analytical Hierarchy Process (AHP) [9], Cumulative Voting, Numerical Assignment, Planning Game and Wieger's method. An example of the second category would be the Win-Win approach.

The prioritization technique that is popular among companies and researchers is the AHP technique. AHP is a multiple criteria decision-making method that has been adapted for prioritization of software requirements [16, 8]. In a software project that has *n* requirements, a decision maker requires $n(n-1)/2$ pair-wise comparisons. Advantages of AHP are that it is trustworthy and has a strong fault tolerance. Along with AHP, another technique called Bubblesort [17] is one of the simplest and most basic methods that have been used for requirements prioritization and is similar to AHP in many ways [18]. However, Bubblesort receives little attention in literature compared to AHP even though both perform well according to other studies [16]. A problem that arises for both AHP and Bubblesort is the scalability. As soon as the number of requirements exceeds twenty, the prioritizing procedure takes too long. However, solutions such as structuring the requirements in a hierarchy of interrelated requirements [16], or applying machine learning techniques [18] have been proposed to overcome this problem.

The variety of prioritization techniques makes it difficult to select the most appropriate one for a specific project situation. Still, the prioritization of requirements proves to be a fundamental part of the responsibilities for a software product manager but it can be a quite demanding, difficult and time consuming task [18, 19]. Even small software product companies that often deal with smaller amounts of product requirements, could benefit from structured techniques to guide them to the best possible software product releases.

1.2 Problem Statement

Concluding from a research of Karlsson et al. [16], Bubblesort and AHP performed the best compared to four other prioritization techniques in terms of reliability, ease of use and fault tolerance when dealing with relatively small amounts of requirements. The reason to center research upon AHP and Bubblesort is because both perform really well according to other studies [16]. Still, there is significantly more literature devoted to AHP. Furthermore, both techniques can easily be applied on a small amount of requirements, which makes it easier to use in an experiment setting [21]. As stated, AHP receives a lot of attention whereas Bubblesort does not, and for this reason, we will discuss both techniques and elaborate on the specific differences between them by providing an experiment with the aim to extend current literature on Bubblesort and AHP. The application of the techniques will be performed by assessing and comparing the results of both techniques through a prioritization experiment with students at a large university in the Netherlands. This experiment will provide specific information about the perceived reliability, time consumption, and usability for both techniques. The research question we want to answer is:

How do the requirement prioritization techniques Bubblesort and Analytical Hierarchy Process perform when compared to each other in terms of perceived reliability, time consumption, and usability?

The next five sections are structured as follows. Section 2 presents the applied research approach and provides a clear explanation for this choice. Section 3 provides information about the execution of the experiment. Section 4 provides a brief explanation of how to interpret the results, and follows with the actual results for the variables tested for each of the techniques. Section 5 includes the overall conclusions. Finally, section 6 provides a discussion, goes into the limitations of this research, and some suggestions for future research.

2 Experimental Design

The goal of this experiment is to compare two prioritization techniques, AHP and Bubblesort, in terms of usability, time consumption and perceived reliability.

2.1 Subjects

For this experiment, we provide twelve students from Utrecht University, The Netherlands, with a list of requirements for a new version of Google Maps. Each subject uses Google Maps regularly. Furthermore, each subject has knowledge of the importance of requirements prioritization and has approximately the same age in order to maintain integrity and diminish variable influences on the results. The subjects did not receive any credits or reward for their participation.

2.2 Objects and Instrumentation

We created a list of twenty requirements[1] for a widely used web-based software package, called Google Maps. Google Maps is a web mapping service application and technology provided by Google. It allows the user to navigate anywhere on earth to view satellite imagery, maps, terrain, 3D buildings and street views. The reason for selecting Google Maps is because it is a well-known web based software product and thus widely-used on a daily basis, causing users to have different experiences and deficiencies about the application.

Each subject is provided with the list of requirements and with Excel based custom-made tools for AHP as well as Bubblesort. Both tools provide the subject with a graphical representation of the algorithm used behind AHP and Bubblesort. Since both tools were made in an Excel environment, any possible influences caused by the spreadsheets can counterbalance each other. Along with the tools, subjects were provided with instructions on how to use both tools.

2.3 Data Collection and Analysis Procedure

Data is collected by means of a questionnaire. The time consumption is registered through identifying the exact starting time and the exact ending time of the subjects excluding any breaks in between. The data is collected in an informal experiment setting at the university, where the subjects use the provided excel based tools along with the questionnaire. When the subjects are finished, the data is entered manually in SPSS (a statistical analysis tool) in order to measure predefined outcomes.

To perform an effective and meaningful comparison of both requirements prioritization techniques, we take three evaluation factors into account: (1) *time consumption*, which is defined as the interval between the time the user starts prioritizing the requirements and the time when the user is finished prioritizing [21] using either of both prioritization techniques; (2) *ease of use*, which is measured by means of direct questions to each participating subject concerning complexity and number of comparisons; and (3) *perceived reliability*, which is defined as the degree to which the final requirements ranking differs from the ideal target ranking. In this research, the ideal target ranking is the ranking the subject has in mind based upon

[1] The list of requirements can be found at http://people.cs.uu.nl/weerd/reew-appendixa.pdf

implicit knowledge. This ideal ranking per subject is the ranking the subject made manually based upon implicit knowledge and without the use of a tool or technique.

2.4 Validity Evaluation

In this experiment, students are used as subjects instead of businessmen; hence reasoning and interpretation of requirements might not be representative for software product companies [21]. Still, the students do have a thorough knowledge about the software product and the presented requirements at hand, and therefore, reasoning and interpretation threats should influence the results only minimally.

On an individual level, there is the risk for students being influenced by familiarity with the requirements or learning from their experience with the first prioritization method upon switching techniques. However, by counterbalancing the groups we tried to minimize this threat. Furthermore, the subjects of the experiment could be influenced by fatigue. Hence, we limited this threat by keeping the requirements understandable and the number of requirements low.

3 Execution

Before starting with the experiment, the subjects were randomly divided into two groups. Each subject that is part of group A will apply Bubblesort on the list of requirements whereas the subjects from group B will apply AHP. When both groups are finished, they will switch techniques, i.e. subjects from group A will then apply AHP and subjects from group B will then apply Bubblesort. The reason for alternating the techniques is to take away any habituation influences of the subjects on the results; i.e. any advantage gained by using one technique before the other for the whole group. In addition, if all subjects start with the same technique, they might be able to influence the results of the other technique due to time constraints or time consumption. When both groups are finished, each subject will fill in a brief questionnaire. This questionnaire encompasses the evaluation of both tools by assigning a grade to the perceived reliability, the usability, and the time consumption of both tools. Afterwards, the results of each subject will be analysed.

4 Analysis of the Results

The results from the questionnaire can be found in Table 1. Each subject that took part in this experiment is assigned to a group. Subjects 1 to 6 were part of group A whereas subjects 7 to 12 were placed in group B. The requirements prioritization process started simultaneously, where group A started with the Bubblesort tool (the abbreviation BS in Table 1) and group B started with the AHP tool. When both groups were finished, each subject had to fill in a brief questionnaire with respect to their experiences towards both tools.

In Table 1, each number in the table represents a grade (on a scale of 1-10) that the subject assigned towards the tool. For instance, subject 1 assigned an 8 to the reliability for AHP and a 4 for the reliability of Bubblesort. In addition to these grades, the consistency ratio is also taken into account for the AHP tool. Since this ratio only applies to the AHP tool, this ratio is not available for the Bubblesort tool. The consistency ratio is presented in the AHP column between the two brackets and describes the consistency of the prioritization process, i.e. how consistent was the subject when prioritizing requirements. According to Saaty [9], a ratio of less than 0.10 is considered acceptable. Furthermore, the time consumption (in minutes) is measured throughout the prioritization process to indicate how much time it took to complete the prioritization process. The results are located within the AHP and Bubblesort columns underneath the overarching 'time consumption' column.

Table 1. Questionnaire results, specified per subject and group

	Reliability		Ease of use		Time consumption		Overall score	
	AHP	BS	AHP	BS	AHP	BS	AHP	BS
Group A								
Subject 1	8(.172)	4	5	3	6(31)	2(59)	8	6
Subject 2	8(.093)	7	7	6	5(66)	6(42)	8	6
Subject 3	7(.054)	9	8	9	8(14)	7(16)	7	8
Subject 4	5(.023)	7	5	9	5(33)	5(29)	5	8
Subject 5	7(.039)	8	6	4	7(29)	6(35)	7	7
Subject 6	7(.040)	8	6	9	5(75)	9(40)	7	9
Average	*7(.070)*	*7,2*	*6,2*	*6,7*	*6(41)*	*5,8(37)*	*7*	*7,3*
Group B								
Subject 7	5(.075)	7	2	4	10(30)	9(20)	4	6
Subject 8	6(.019)	9	7	9	4(60)	9(30)	6	8
Subject 9	6(.144)	8	7	9	5(57)	8(30)	6	8
Subject 10	8(.152)	10	6	9	6(24)	9(18)	6	9
Subject 11	7(.061)	7	7	8	3(25)	6(13)	6	7
Subject 12	8(.061)	4	5	6	4(30)	5(30)	6	4
Average	*6,7(.085)*	*7,5*	*5,7*	*7,5*	*5,3(38)*	*7,7(24)*	*5,7*	*7*
Average	**6,83**	**7,33**	**5,92**	**7,08**	**5,67**	**6,75**	**6,33**	**7,17**

In addition to Table 1, a graphical overview is presented in the form of a bar graph in Figure 1. In the following subsections, we will elaborate upon each measurement.

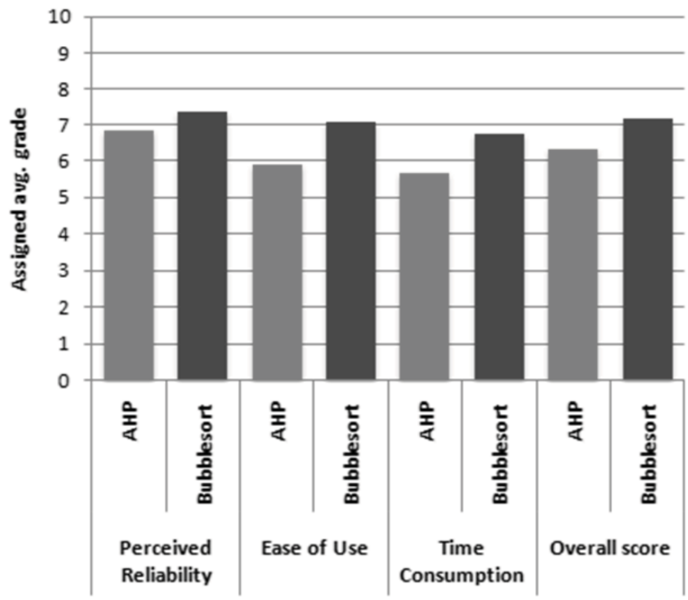


Figure 1. Graphical overview of the results, divided per measurement.

4.1 Perceived Reliability

The most left column of Table 1 shows the obtained reliability scores from the subjects involved in the experiment. These scores, on a scale of 1 (low) - 10 (high), were assigned by each subject based on their consensus with the actual ranking obtained through each technique. Furthermore, these results should be considered as 'perceived' because the results are based on the basic reasoning process of non-actual stakeholders in the development of this Google Maps release. When testing for significance between both groups A and B, using the Mann-Whitney Test, there proved to be no significant difference between both groups (α =.05). Therefore, group A did not score significantly better than group B en thus we can merge both group scores and assess the full group result.

Before continuing with the evaluation of the reliability scores, the consistency of the AHP technique needs to be taken into consideration. As can be seen from table 1, the contents of the brackets, next to the assigned reliability score for AHP, provide the Consistency Ratio (CR) for the valuations provided by the subjects. This consistency ratio is composed of the consistency index and the random index [8, 20]. Any CR lower than .10 is considered acceptable, however, according to Saaty [9] these 'acceptable' scores are hard to obtain. Looking at the results, consistency proves to be very good with only three people obtaining CR-scores higher than .10. For one particular subject we observed a remarkable consistency rate in AHP for filling in the matrix cells, indicating a very inconsistent valuation (.172) of the requirements, while his appraisal of the obtained ranking (8) was quite high. For that particular subject we can conclude that by providing a very inconsistent valuation of the requirements, his results should be considered unreliable.

Using the Wicoxon Signed Ranks test, the overall deviation between the scores assigned by the subjects for AHP (avg. 6.83) and Bubblesort (avg. 7.33) proved not to be significant (α =.05). Nevertheless, the difference between the mean of the scores of the perceived reliability for the two techniques suggests that Bubblesort provides better results on 'perceived' reliability.

4.2 Ease of Use

Ease of use (i.e. the usability) is measured through the use of a questionnaire. The subjects were kindly asked to rate the usability of Bubblesort and AHP on a scale from 1-10, where 10 is the highest score.

When calculating the average score of the usability of both tools, we see that Bubblesort has a mean of 7.08, whereas AHP does not reach beyond 5.91. This result is as we expected since the Bubblesort tool required less time to complete, and is therefore regarded as easier to use. This conclusion could also be derived from the fact that group A started with Bubblesort first. Since group A did not have a clue how the AHP tool looked like, they gave Bubblesort an average of 6.7. Group B, which started with AHP first, gave Bubblesort an average grade of 7.5. Since the tool of AHP perceived to be more complex in use, the subjects subsequently found the Bubblesort tool easier to use and thus assigned a higher grade to it. The usability score of both techniques could be influenced by the created Excel-based tools. During the prioritization process, some subjects faced difficulties with respect to the completion of the Excel document whereas other subjects did not face any troubles. Nonetheless, considering both tools were made with Excel, any possible drawbacks caused by the limitations of spreadsheet software during the prioritization process would likely counterbalance each other, thus negating their effect.

Analyzing the usability between group A and group B, based on the Mann-Whitney Test (α =.05) we again did not find a significant difference between both groups. Since no significant difference was found, we decided to merge both groups A and B. When we subsequently use the Wilcoxon Signed Ranks test to test for significant differences between the scores of AHP and the scores of Bubbelsort, we again did not find any significant differences (α =.05). Although the average scores deviated, based on our sample size, this deviation cannot be assigned to the population yet. There is only a strong presumption that Bubblesort might be easier to use than AHP.

4.3 Time Consumption

The averages in Table 2 show that Bubblesort performs better compared to AHP in terms of the actual time consumption. The results from start/end time were recorded by the subjects. As illustrated in Table 2, the difference in time consumption (sample mean) of the two prioritization techniques is 9.3 minutes, which is relatively large.

Even though both medians are nearly the same (30.5 for AHP and 30.0 for Bubblesort); the difference in time consumption can easily be noticed when taking standard deviation into account. In our experiment, the distribution of the actual time consumption is not skewed and there is no extreme outlier as well. Hence, we can conclude that the Bubblesort tool is more time saving compared to the AHP tool.

Aside from measuring time consumption in minutes, we use the time consumption score, which is a subjective judgment given by the subjects after they use both techniques. The higher the score, the less time consuming the tool is. As shown in Table 2, Bubblesort achieves a higher score, which means that the subjects believe Bubblesort is less time consuming. The mean score that Bubblesort obtains from the subjects is 6.7, which is higher than the 5.7 of AHP, and the median of the scores for these two techniques are 6.5 and 5, respectively. Although, according to the standard deviation, the scores obtained by Bubblesort are more dispersed than AHP, this does not change the fact that there still is an indication Bubblesort is a better technique with respect to time consumption.

Table 2. Time consumption in minutes and grade per tool.

Method	Median	Mean	Std. deviation(SD)
AHP (in minutes)	30.5	39.5	19.5
Bubblesort (in minutes)	30	30.2	12.9
AHP (in grade)	5	5.7	1.9
Bubblesort (in grade)	6.5	6.7	2.2

4.4 Overall Rating

After taking reliability, ease of use, and time consumption into account, we also asked the subjects to assign an overall score to both techniques. This was again based on a scale of 1-10; the higher the score, the higher the appreciation of the technique.

If we look at the overall scores in Table 1, we notice a higher score for Bubblesort in comparison to AHP. This is coherent with the other scores obtained for reliability, ease of use, and time consumption, because those also indicated better averages for Bubblesort. The average score for AHP is 6.33, while Bubblesort shows an average score of 7.17. Furthermore, subjects were asked to explain their overall score and indicated that the combination of the quickness and the reliability of Bubblesort, in comparison to AHP, were the main reason for assigning higher scores to the former.

Analyzing the overall scores for both techniques within group A and group B, based on the Mann-Whitney test, did not show any significant differences (α =.05) when comparing for Bubblesort but there was a minimal significant deviation between group A and B for AHP. This is considered to be quite remarkable since the overall score should be derived from the reliability, ease of use, and time consumption. Still, merging of both groups was difficult and a general comparison would be less reliable. Nevertheless, overall comparison, although not encouraged here, showed no significant deviation between Bubblesort and AHP.

5 Conclusion

Throughout this research, we attempted to provide an answer to the following research question:

How do the requirement prioritization techniques Bubblesort and Analytical Hierarchy Process perform when compared to each other in terms of perceived reliability, time consumption, and usability?

For this research we conducted an experiment among twelve Master students of a large University in the Netherlands to test the applicability of Bubblesort and AHP. The subjects used both techniques in prioritizing twenty fictional requirements for a new software product release of Google Maps. The perceived reliability, ease of use and time consumption was assessed by comparing both techniques. Taking both groups into account, there were some distinct differences found.

On average, Bubblesort scored better than AHP in all aspects of the comparison. The perceived reliability, the time consumption and the usability of Bubblesort was greater than AHP. Nevertheless, the division of both groups did have one minor implication on the results: the overall valuations of the tools, assigned by the subjects, seemed to be influenced by the group division.

During the experiment we tried to eliminate any advantages gained from using one of the tools before the other. When looking at the results for both groups, we saw no further significant differences between both groups ($\alpha = .05$). Therefore, we can conclude that both groups scored the same on both techniques.

After merging both groups to create an overall average, we can conclude that Bubblesort still scores better than AHP for perceived reliability of the results. Furthermore, Bubblesort also scored better on time consumption and ease of use by a decent margin. Unfortunately, the sample size proved to be an obstacle in obtaining significant results.

Concluding from our research, Bubblesort scores overall better compared to AHP. There is a strong indication that when the sample group size will be increased, Bubblesort would probably still outpace AHP when both techniques are applied in a similar situation.

6 Discussion, Limitations, and Further Research

Throughout our experiment we measured the perceived reliability, ease of use, and time consumption for both requirement prioritization techniques AHP and Bubblesort. However, the research presented in this paper has certain limitations. These limitations predominantly originate from the size of our research sample and the unavailability of established tools for both techniques.

Firstly, due to a lack of availability of subjects as well as time constraints, we were not able to increase the size of the sample. The time provided to assemble a representative large group of subjects proved to be limited. Therefore, the reliability of the obtained results can be influenced. Upon further review, the small sample size was not considered to be a normal distribution and non-parametric tests had to be performed. However, the results seem to give a decent indication of expected results when the sample size is increased. Therefore, we would suggest further research with an increased sample size.

Secondly, the unavailability of a usable and implementable tool for both techniques could have influenced the results. Currently, there is no established tool for Bubblesort available. Therefore we developed a custom Excel-based tool for Bubblesort fitting the number of requirements used within our experiment. Furthermore, we tried to use the IBM© focal point™ tool for AHP but encountered some problems with implementing it. Therefore, we developed a custom Excel-based tool for AHP too. Although this eliminates any problems on comparing the two techniques, there is a possibility that the tools were not comprehensive and sufficient enough due to a lack of validation of the tools. Since the participating subjects did not experience any difficulties nor did we hear any complaints, we believe the tools were sufficient enough to obtain reliable results on perceived reliability. However the results of this experiment on time consumption and ease of use will be very tool dependent and therefore specific for our custom tools.

A final note should be made on the issue of prioritizing large volumes of requirements. In this experiment, a set of twenty requirements was used. For software vendors dealing with small amounts of requirements, Bubblesort may be a good solution to prioritize requirement and reach consensus among the stakeholders. However, in many real-life settings, the amount of requirements easily exceeds the amount of requirements we used. Further research should be done to tool support in which Bubblesort is integrated with proposed solutions as structuring the requirements in a hierarchy.

References

1. Bebensee, T., Weerd, I. vd., Brinkkemper, S.: Binary Priority List for Prioritizing Software Requirements. In: Requirements Engineering: Foundation for Software Quality. LNCS, vol. 6182/2010, pp. 67--78. Springer Heidelberg (2010)
2. Wiegers, K.: First Things First: Prioritizing Requirements. Software Development 7(9), (1999)
3. Kabbedijk, J., Brinkkemper, S., Jansen, S.: Customer Involvement in Requirements Management: Lessons from Mass Market Software Development. In: 17th IEEE International Requirements Engineering Conference, pp. 281--286. Atlanta (2009)
4. Lehtola, L., Kauppinen, M., Kujala, S.: Requirements Prioritization Challenges in Practice. In: Proceedings of 5th International Conference on Product Focused Software Process Improvement, pp. 497-508. Japan (2004)
5. Lehtola, L., Kauppinen, M.: Empirical evaluation of two requirements prioritization methods in Product Development Projects. In: Proc. European Software Process Improvement Conference, pp. 161-170. Trondheim, Norway (2004).
6. Bekkers, W., Weerd, I. van de, Spruit, M., Brinkkemper, S.: A framework for process improvement in software product management. In: A.Riel et al. (Eds.): EuroSPI 2010, CCIS 99, pp. 1--12 (2010)
7. Weerd, I. van de, Brinkkemper, S., Nieuwenhuis, R., Versendaal, J., & Bijlsma, L.: Towards a reference framework for software product management. Proceedings of the 14th International Requirements Engineering Conference, Minneapolis/St. Paul, Minnesota, USA, 319-322 (2006)

8. Karlsson, J., Ryan, K.: A cost-value approach for prioritizing requirements. IEEE Software 14(5) 67--74 (1997)
9. Saaty, T. L.: The Analytical Hierarchy Process, McGraw-Hill (1980)
10. Berander, P., Jönsson, P.: Hierarchical Cumulative Voting (HCV) - prioritization of requirements in Hierarchies. International Journal of Software Engineering and Knowledge Engineering 16(6), 819--849 (2006)
11. Karlsson, L, Thelin, T., Regnell, B., Berander, P., Wohlin, C.: Pair-wise comparisons versus planning game partitioning – experiments on requirements prioritisation techniques. Empirical Software Engineering 12(1), 3--33 (2007)
12. Karlsson, J.: Software requirements prioritizing. In: Proceedings of 2nd International Conference on Requirements Engineering, pp. 110–116 (1996)
13. Perini, A., Ricca, F., Susi, A.: Tool-supported Requirements Prioritization: Comparing the AHP and CBRank Methods. Information and Software Technology 51(6), 1021--1032 (2009)
14. Racheva, Z., Daneva, M., Buglione, L.: Supporting the Dynamic Reprioritization of Requirements in Agile Development of Software Products. In: Proceedings of the Second International Workshop on Software Product Management 2008, Barcelona, pp. 49--58 (2008)
15. Berander, P., Andrews, A.: Requirements prioritization. In: A. Aurum, C. Wohlin (Eds.), Engineering and Managing Software Requirements, Springer, (2005)
16. Karlsson, J., Wohlin, C., Regnell, B.: An evaluation of methods for prioritizing software requirements. Information and Software Technology 39(14--15), 939--947 (1998)
17. Aho, A. V., Hopcroft, J. E., Ullman, J. D.: Data structures and Algorithms. Addison-Wesley, Massachusetts (1983)
18. Avesani, P., Bazzanella, C., Perini A., Susi, A.: Facing scalability issues in requirements prioritization with machine learning techniques, In: Proceedings of 13th IEEE International Conference on Requirements Engineering, IEEE Computer Society, Paris, France, pp. 297–306 (2005)
19. Firesmith, D.: Prioritizing Requirements. In: Journal of Object Technology 3(8), pp. 35--47 (2004)
20. Random RI for AHP, http://www.docin.com/p-23136057.html
21. Berander, P.: Using Students as Subjects in Requirements Prioritization. In: International Symposium on Empirical Software Engineering (ISESE'04), pp. 167--176 (2004)

Speed Creation Session: A way to increase the productivity of experts in projects and assure quality requirements

Matthias M. D. Pohle[1], Sven Krause[2] and Andreas Rusnjak[3]

[1] Swisscom (Switzerland) Ltd, Corporate Business P.O. Box, CH-3050 Berne, Switzerland
[2] Zühlke Management Consultants Ltd, Wiesenstrasse 10a, CH-8952 Schlieren, Switzerland
[3] Christian-Albrechts-Universität zu Kiel, Department of Informatics, 24118 Kiel, Germany

matthias.pohle@swisscom.com, sven.krause@zuehlke.com, aru@informatik.uni-kiel.de

Abstract. In large and distributed companies the knowledge is spread. Experts come from different locations and subsidiaries. People often work in several projects simultaneously. This leads to an efficiency loss and long product development cycles. What if there is a way to get rid of dozens bilateral and specialized division workshops and preparation meetings? We call our approach "Speed-Creation". The starting point is fuzzy frontend of innovation, where the rough product or software idea is given. But still it`s vague and fuzzy. At this point the experts ask "What are your requirements and what is the impact on my field of work?" But these answers don`t exist at this point. So we set up a small team being led by a speed-coach. During 72 hours they only work on this single project without distraction. After this seed phase a feasibility study can be easily set to prepare a development project and ultimately production of an innovative product. This paper describes the Speed Creation approach and reports on experiences of applying the method.

Keywords: Project Management, Product Management, Product Development, Innovation Management, Early Phases, Business Analysis, Business Engineering, Agile Development, Team Building, Team Excellence.

1 Introduction

In today's time we live in a world of information overflow. A lot of development projects have huge scopes and big teams. Often project leader work together with 40 - 50 or even more project delegates and they do not know each other and have never worked together before. The early phase of these kinds of projects is often critical and inefficient. It takes some time to get all the resources and then these 40 to 50 highly specialized delegates rattle the project leader with dozens of specific questions and asking about detailed requirements (see Fig. 1). But the project is in an early phase; the project idea is still fuzzy. Product managers analyzed the idea well. Product managers got the needs, developed an approach and thought about the benefits and if the approach could be substituted by others. So they are on a good path and have made great analytics. But the implementation approach is very general at this time; it is just a rough idea and detailed requirements, e.g. for an ensuing feasibility check of the project and state compliance are still missing.

Due to this the Speed Creation will be introduced as a lightweight process model to gather a lot of substantial requirements by an interdisciplinary group in a short time.

But where is the difference to other creativity groups, e.g. the Focus Groups? Yes, other creative methods proceed similarly and contain many similar components, but why invent something new, if there is an existing working process or similar approach? Nevertheless there are differences. The first difference is focus. The speed creation focuses itself on the requirements that are needed for the implementation. That means the question "What is needed to realize the product?" will be answered. Speed Creation builds that bridge from a rough idea to an 80% draft of business requirements to allow a project team to start with a fundamental base of requirements into a feasibility study. It is a new agile project method that combines the best elements from other methods, for example scrum, kaizen and customer experience design.

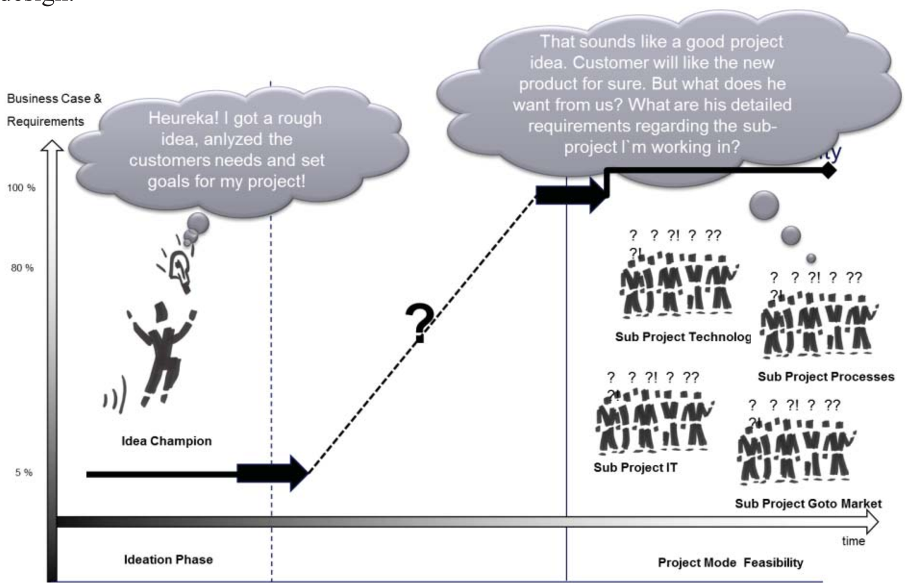


Fig. 1. The problem of exploration projects is showing the gap in the early project phase.

2 Speed Creation process

Speed Creation helps to increase the efficiency of requirements elicitation and consolidation. The method is introduced by describing necessary preparations, how Speed Creation is performed, and how its results then can be utilized (c.f. Tab. 1).

The starting point is an idea. This idea can be developed within an idea workshop [1], a process for generating innovation candidates [2] or any other approach. If you have a good idea, move on to the Speed Creation mode. To ensure that only good ideas make it to Speed Creation, we typically apply a two-step filter. The idea is summed up on one page. During this process, you will describe in a high abstraction the needs of the customer, your approach, the benefit for the customer and your company and the situation of your competitors. This method is called NABC (Need, Approach, Benefit and Competition) [3, 4].

The product manager is responsible for filling out the NABC. This NABC is assessed by a selected management team. Criteria for the approval are the compliance to the product strategy, to the product portfolio and project management. After a positive decision, the Speed Creation can start.

Speed Creation process overview	
Goal	Increase of the productivity of the project members Speed up projects
Precondition	An approved idea for a product or a service development
Steps	1) Workshop preparation and set up a small and interdisciplinary project team
	2) Deepening and concretization of the project order (prepare ecosystem, goals, scope)
	3) Create a first view of the offering
	4) Specify use cases along the "customer experience chain"
	5) Define general requirements, requirements for others (like other products or projects)
Post condition	• 80% of the business requirements, which is the basis of the ensuing feasibility study • A video where the product manager (or the project team) presented the customer needs and product goals • A motivated core team with a common understanding • A cultivated cross-organizational teambuilding

Tab. 1. Speed Creation - Overview

There is a fixed set for workshops, analyzing and documentation methods we use. Every Speed Creation can be somewhat different, hence needs to be planned individually. That is one of the key success factors. In Speed Creation we set up a small and interdisciplinary project team (see Fig. 2). It should not be bigger than 5-7 people. We have a rough idea and start to fill white pages of paper with requirements. And in a creation workshop the critical mass is round about 7 people. You can easily do a review with 10+ people, but you can hardly create new coherent content with big groups. So we set up this interdisciplinary team and focus on an intense 72 hours' workshop. The interdisciplinary team consists of product manager, marketing manager, sales people, business analyst and IT or technical people. Within this workshop we analyze and concretize the idea in depth. But it is still in best guess mode. We create the input which the experts will refine later.

When starting the Speed Creation we look at the ecosystem (on the market as well as internally, e.g. it-systems) first. Then we set goals for the outcome of the speed creation. Due to this it is also very important to specify the scope of the idea which will be developed in a Speed Creation. The objective of this phase is to bring the needs with the goals and the scope in agreement.

A matrix ensures that each goal covers at least one need. Additionally, a second matrix ensures that each scope element covers at least one goal. That gives us the certainty that we cover all needs in further considerations. With this we ensure that we do not observe goals or scope elements which have no impact on the customers' needs. Thereby we ensure that the stakeholders do not exploit the situation for their own goals.

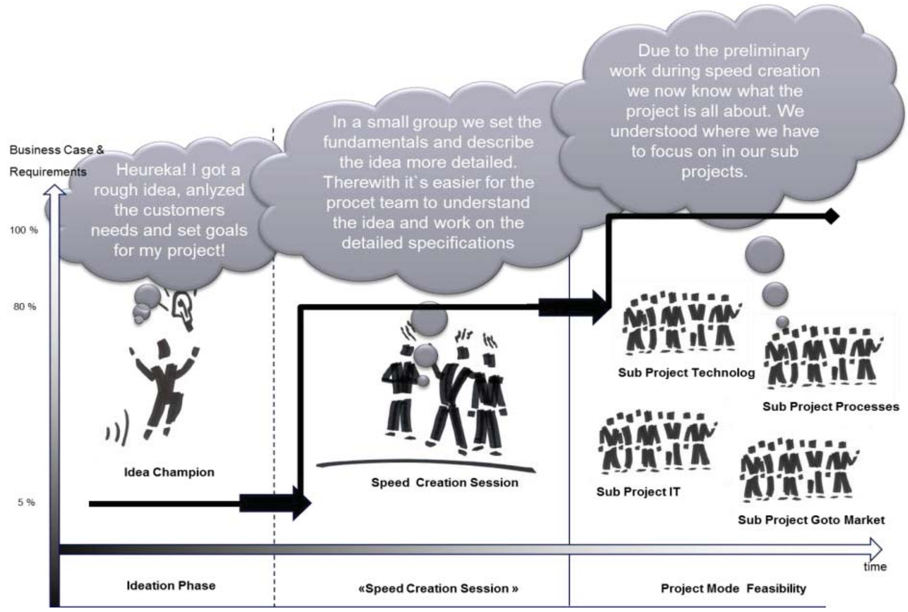


Fig. 2. The “new” story showing how speed-creation closes the gap in the early project phase.

In a workshop we create a draft for the offering to the customer. Later we specify a first draft of use cases along the so called customer experience chain. In this process we use the typical use case notation based on UML and the table-specification after Cockburn [5]. The customer experience chain separates the customer interactions with the enterprise into specified phases. The 8 phases are notice, inform, order, take in operation, use, pay, get support and offer change.

The documentation within a Speed Creation is much hands-on and must be done very quickly (on the fly) during the teamwork, e.g. writing requirements into a word document, taking snapshots of drawings on whiteboards/ flipcharts. But hands-on does not mean a bad quality of requirements. It only means that missing details have to be later supplemented or out-formulated. The requirements described are based on a phrase pattern [6] in order to ensure understandability. Basis of the documentation is a structure similarly to Volere [7] and IEEE 1233-1998 [8]. The structure contains the chapters’ situation analysis (customer needs, goals, and scope), business case (cost effective study), commercial offer, use cases, general requirements, and requirements for others (like other products or projects).

The outcome of a Speed Creation contains the content of what the requirements are. Maybe some details are missing, but all the 40 to 50 project delegates who are involved in the project, later in the feasibility phase, get a good overview on what is planned on a more detailed basis. Within the ensuing feasibility study the delegates now know what to work on; which problems they need to solve and where to dig deeper into their expertise. They then add the detailed requirement specifications.

The speed philosophy claims to improve projects by focusing: Take one piece of work at a time. Concentrate on one topic and then move on to the next. Do not do too much of parallel work streams. Second is to work out of office where none of your daily colleagues can distract you from the project work. Also it is nice to work in a pleasant environment maybe at a lakeside. Third you got a moderator who is a professional on how to product development. The expertise regarding the specific product will not come from method knowledge, but the moderator brings in the method knowledge.

3 Specifics to the Speed Creation process

A good time management especially during Speed Creation workshops is not easy but important. You have to define time blocks for each topic area and communicate the existing time to execute the topic area. It is important that the persons involved can focus themselves on the essence of the topic. Based on a clear structure given by the Speed Creation document or prepared flipcharts which helps you to save time, the moderator needs a good feeling for the situation. The moderator has to decide whether he would like to adapt the given time. But, the moderator has to ensure that the duration for the other topics is sufficient.

During the workshop we use flipcharts and drawings. The pictures get pasted into the document. It is only a first draft to refine on later. There is no need to draw it exactly in a software tool. There is time in a speed-creation. Nevertheless every picture is described in prose by the speed-documenter.

4 Lessons Learned

There are four big wins if you choose to use a speed-creation in your project. You will speed up your projects, you create a common understanding and conserve it in the documentation, you ease feasibility and you cultivate cross organizational teambuilding.

Projects will be faster and more efficient because you got a detailed first guess of requirements within 72 hours and not 5-6 months. The Modus operandi comprises a good time management for the different workshops, including preparation and documentation. For a solid time management, professional preparation is important.

The common understanding gets more important the bigger the project team gets in the later phases, e.g. the feasibility or implementation phase. The concentrated work we create at the same location facilitates a common understanding. One can achieve a common understanding by supporting discussions and exchange opinions. Naturally is happens formally during workshops, but also informally during e.g. dinner. The importance of an informal exchange outside the scheduled workshops is not to be underestimated. So for that reason, Speed Creation workshops should be held outside the office including overnight stays in hotels.

The ensuing feasibility project will be easier. All the delegates who come into the project later on to provide expertise on specific topics now know exactly what to do. The frame is set; they can pick up on the existing requirements and fragments and detail them any further.

Due to the fact that the speed-creation core-team is multidisciplinary we tear down walls between different organizations and lead the people into a teambuilding phase just for the product. This works well because of the 72 hours approach. Within that time people work hard and play hard. So they start to be a team.

5 Key Success Factors

There were six key success factors which determine to have an efficient Speed Creation. First there was expectation management. How concrete or fuzzy was the idea itself? Was it clear what the idea is? Were the goals set? If not, you could hardly have a good Speed Creation. Second was to customize Speed Creations. Each project differs, so the detailed workshop plan needs to be individualized. 80% are mostly identical, but some parts just need to be individualized. Factor three and four were the concentrated work without distraction and to get feedback from a jury. During a Speed Creation people were so focused that they sometimes tend to be over focused. Therefore we presented what they have done during the day to a jury. This jury consists of colleagues who did not take part on the workshop process. They get the presentation and give feedback. One feedback should be positive (what they liked in the presentation), another should be negative, but please stay constructional. Factor five was having a multidisciplinary Speed Creation team. And finally factor six the product manager and/or business engineer. During a Speed Creation the product manager and/or business engineer might focus on his role and concentrate on the discussion. Everything else is up to the two speed coaches who moderate and also document. But after a Speed Creation the product manager has to take the lead and move on with his business case and the business requirements set. The outcome from the Speed Creation is still an 80% draft and it needs to get finalized.

References

1. Maiden, N., Gizikis, A., Robertson, S.: Provoking Creativity: Imagine What Your Requirements Could Be Like. IEEE Software 21, 5 (2004) 68-75
2. Gorschek, T., Fricker, S., Palm, K., Kunsman, S.: A Lightweight Innovation Process for Software-Intensive Product Development. IEEE Software 27, 1 (2010) 37-45
3. Carlson, C., Wilmot, W.: Innovation: Five Disciplines for Creating What Customers Want. Crown Business (2006)
4. A shortened English version of Kirstine Vinderskov's work paper 'NABC – metoden, www.learnship.eu/media/2657/dk%20nabc.doc
5. Cockburn, A.: Writing Effective Use Cases. Addison-Wesley Longman (2000)
6. Cohn, M.: User Stories Applied: For Agile Software Development. Addison-Wesley Longman (2004)
7. Robertson, S., Robertson, J. (1999). Mastering the Requirements Process. Addison-Wesley
8. IEEE (1998). IEEE Recommended Practice for Software Requirements Specifications. IEEE Standard 830-1998

4 Requirements Prioritization for customer-oriented Software-Development (RePriCo)

Editors

Benedikt Krams
Universität Stuttgart, Germany, krams@wi.uni-stuttgart.de

Sixten Schockert
Universität Stuttgart, Germany, schockert@wi.uni-stuttgart.de

Technical Programme

Second Workshop on Requirements Prioritization for customer-oriented Software-Development RePriCo'11 An Introduction

Georg Herzwurm[1], Wolfram Pietsch[2]

[1] Department for Business Administration and Information Systems, esp. Business Software, Universität 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 and workshop content

RePriCo'11 represents the Second Workshop on Requirements Prioritization for customer-oriented Software-Development (RePriCo'11) held at the 17th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ2011).

As far as prioritization is an essential task within requirements engineering in order to cope with complexity and to establish focus properly two perspectives can be identified:

- From a formal standpoint of view prioritization 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 therefore 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.

We are glad about holding RePriCo'11 for the second time at REFSQ in Essen: in 2010 ambitious participants from research as well as industrial practice discussed two full research papers and four position papers in an open-minded and pleasant atmosphere.

In 2011 RePriCo'11 attracted 9 submissions. Each submission was reviewed by two members of the program committee. Based on the reviews, four submissions were accepted as full research papers and one submission as short paper.

The submissions comprise current research findings from various fields: IT risk estimation based on qualitative empirical results (Andrea Herrmann); prioritization of requirements and customer segments for software product lines and variants (Andreas Helferich, Lars Oliver Mautsch); concretization of requirements by a knowledge-based and computer-aided method (Constanze Kolbe, Robert Refflinghaus); formal prioritization methods and presentation as well as evaluation of one method implemented in industrial practice (Annabella Loconsole, Hannes Gruber, Adrian Nae, Björn Regnell); furthermore one approach for IT supported optimization of business process chains between public administration and business (Norman Riegel, Oezguer Uenalan, Thomas Jeswein).

Results of our workshop evaluation (questionnaires filled out by all attendees) showed, apart from a positive overall evaluation of the workshop, that especially the variety of research findings in the five presentations pleased all participants.

We are convinced that the workshop was rewarding alike 2010 and findings in these proceedings encourage researches as well as software-developers, requirements engineers or consultants to absorb new ideas and carry them out into their daily work and research projects.

Our special thanks go to all speakers and participants for their contributions to the workshop. We are confident in organizing RePriCo in 2012 as well and looking forward to meeting you in Essen again.

2 Organization

2.1 Program Committee

Chair

Prof. Dr. Georg Herzwurm, Universität 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
Priv. Doz. Dr. Andrea Herrmann, Axivion GmbH, Germany
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
Priv. Doz. Dr.-Ing. Robert Refflinghaus, TU Dortmund University, Germany
Dipl.-Wirt.-Inf. Sixten Schockert, Universität 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, Universität Stuttgart, Germany
Dipl.-Wirt.-Inf. Sixten Schockert, Universität Stuttgart, Germany

Information Need of IT Risk Estimation – Qualitative Results from Experiments

Andrea Herrmann

Axivion GmbH, Nobelstr. 15, 70569 Stuttgart, Germany
AndreaHerrmann3@gmx.de

Abstract. IT risk estimation is a basic activity supporting risk-based requirements prioritization, but also risk-based testing and risk management. Experiments participants experience IT risk estimation as difficult, estimated risks vary strongly among participants and estimations are badly repeatable. One main explanation for the difficulty and uncertainty in IT risk estimation is that much information is needed. This publication discusses what information is needed for IT risk estimation. Such information need was identified and classified by a Grounded Theory analysis of qualitative experiment results. Knowing what information is needed helps to prepare estimation workshops.
Keywords: requirements prioritization, IT risk, risk estimation, software risk

1 Introduction

Risk-based requirements prioritization ranks requirements regarding their potential to reduce risks. For instance, one can estimate the same risk twice – once assuming that the requirement is implemented and once without [AHP04], [HP09]. If, for example, a certain usability requirement reduces the risk of a user error from an estimated 10 times to 4 times per month and the cost of this risk is 10€, then the requirement reduces risk by 6 x 10€ = 60€/ month. This amount is to be compared to the requirement´s cost as well as to the risk reduction achieved by the other requirements. Independently of which method or formula is used for transforming IT risk estimations into requirements priorities, the IT risk estimation task is the basis for all conclusions and decisions built on it. Similarly, risk estimations are used for other kinds of decision-making during software engineering, e.g. by a tester for risk-based testing (prioritizing test cases and errors) or by the project manager for risk management. Together with cost and benefit estimation, risk estimation can be said to be one of the fundamental tasks which provide for quantitative data which influence many decisions. Because the output of the IT risk estimation task is potentially used for success-critical decisions, it must be as correct as possible.

It has been shown by many researchers that risk estimation is difficult. In six experiments, we investigated more in depth with respect to which criteria IT risk estimation is difficult and how IT risk estimation can be improved. This publication analyzes qualitative findings (i.e. free text answers) from three of these experiments in order to investigate one main explanation for difficulty and uncertainty in IT risk estimation: We found that a lot of information is needed to estimate risk.

Which information this is we conclude from the experiment participants´ comments. We analyzed these comments using the Grounded Theory method.

The remainder of this publication has the following structure: Section 2 gives an overview on related work. In Section 3, the three experiments are presented which produced the data which is analyzed in this publication. The Grounded Theory is introduced in Section 4. Section 5 presents the results we received when applying Grounded Theory to the qualitative data (i.e. free text comments) from the experiments. These results answer the question which information is needed for IT risk estimation. Section 6 discusses validity and Section 7 summarizes the results and draws conclusions.

2 Related Work

This related work section refers to two types of work: (1) work about how risk estimations can be used for requirements prioritization, and (2) what is difficult about IT risk estimation and what are Best Practices which improve it.

A *risk event* is an event which could take place in the future with a certain probability and which might have unwanted consequences. One risk event usually is not sufficient to describe what might happen. Instead, the security engineering often uses scenarios for describing risk, for instance in the form of Misuse Cases. Traditionally, Misuse Cases are used to elicit security requirements [SO00], [SO01]. However, MOQARE [HP08] successfully applies Misuse Cases to all types of non-functional requirements. Then, these Misuse Cases describe unwanted scenarios, where the misuser is not always a malicious attacker, but might be a user who by mistake impairs data integrity, or a developer who by negligence threatens the maintainability of a software. Misuse Cases during requirements elicitation can be used to identify countermeasures, i.e. requirements which, if satisfied, prevent, mitigate or detect Misuse Cases and thus support the satisfaction of security and of other non-functional requirements. Countermeasures can be requirements on the IT system, on its architecture, its development process, operation environment or personnel.

Risk r quantifies the importance of a risk event or Misuse Case by multiplying its probability p with the severity of the consequences (or, damage d): $r = p \cdot d$ [ISO02].

When requirements are prioritized based on risk estimations, then usually it is estimated by how much the countermeasure´s implementation reduces a certain risk. This means to estimate risk twice for a system where this requirement is implemented and another system where it is not. This principle is used especially in the context of security (as in [AHP04]), but makes sense also with other non-functional requirements [FCK06], [HP09]. The Failure Mode and Effects Analysis (FMEA) [Sta03] and WinWin [PPB99] also apply it to functional requirements.

From literature and from our own experience, we have gathered the following difficulties of risk estimation and the corresponding recommendations:

Need of a reference system: Risk estimations must refer to the same reference system in order to be comparable [HP09], which is an idea of a system (in terms of which requirements are supposed to be satisfied).

Risk estimation demands *quantitative data about the past*. People have enormous difficulties in judging afterwards how often something happened [TK74], [BF00], [RRM02]. Good estimations need training and short feedback loops [BF00], [BF04]. There are three ways how to obtain quantitative data: measurement of historical data, public risk statistics (like [CER06], [Ric03]), or expert opinions [FC03].

Things change and risk estimation demands predictions about the future. You want to quantify risk for risk events which take place in the future, in a changed world and you do this before the new software system is implemented.

Do you *ask the right person*? Expert knowledge is needed from different perspectives, e.g. management knowledge, user knowledge, and technical knowledge.

Quantifying risk by *the product of probability and damage* does not allow to distinguish between high probability low damage risks and low probability high damage risks [BSI05], which you might want to treat differently. Therefore, always document the estimations of probability or damage.

Complexity: Feather et al. [FCG01], [FCL00] emphasize the importance of tool support, e.g. for visualization. For visualization, they use cost-benefit-diagrams or diagrams which for each risk present likelihood and impact on the two axes.

Time need and cost-benefit-ratio: The time need for risk-based requirements prioritization is higher than for other prioritization methods, because more estimations are to be made for performing the same task (i.e. estimate both probability and damage twice) and also each single estimation was found to take more time [HP09]. Effort-saving strategies can be: Estimate only those risks which are difficult to judge [Dav03] or which support the decisions of strategic importance [How68], reduce complexity by grouping similar risks into categories [REP03], [AHP04], and estimate risk at the point of time when it is needed.

3 The IT Risk Estimation Experiments

This publication analyzes qualitative findings (i.e., free text comments) from three experiments about risk-based requirements prioritization. Our objective is to explain why risk estimation was experienced as being difficult, why the risk estimations varied among participants and also varied when the same person repeated the same estimations several times. Especially, our research question is which information is needed for doing IT risk estimations. We search the answer to this question in the questionnaires analyzing the answers to the question what assumptions the estimators made when estimating risk and in comment fields concerning reasons for uncertainty and difficulty of IT risk estimation. While quantitative results have already been published [HP09] or will be published separately, here we summarize the qualitative results from three of our experiments, i.e. from 46 questionnaires.

These experiments are described in the remainder of this section. What is common to all three experiments is that risks where described as risk scenarios (Misuse Cases), specifying misuser, vulnerabilities, executed threat, and damage caused (like in MOQARE [HP08]). The participants knew MOQARE and its concepts.

The *first experiment* has been described and published before as “Experiment 1” in [HP09].

Objective: The experiment investigated practical needs of risk estimation, e.g.: time need, knowledge and material needed, the influence of group discussions. We also compared risk estimation to a simpler requirements prioritization method.

Sample population**:** Ten master students of the University of Heidelberg in the winter term 2006/07. They had been taught prioritization methods and risk-based requirements prioritization in a lecture before.

System: The experiment used twelve risks and nine requirements from a case study discussed in the lecture and in preceding homework: an Internet flea market.

Material: The material was an introduction (describing the objective of the experiment and the case study, including information about the company, competitors, project execution and staff, and the flea market´s functionalities), the two questionnaires supporting the two prioritization methods, a questionnaire which asks the participants to rate the methods, and questionnaire 5.

Execution: The experiment was performed in a three hour session. Before the first questionnaire was distributed, there was an introductory presentation which explained the objective of the experiment and the methods, as well as the case example. During the experiment, the students individually estimated risks on paper questionnaires.

Main results: Moderated group discussions have positive effects (compared to individual estimations), although they are time-consuming. Risk estimation is difficult and requires a lot of information. Providing statistics to the estimators influenced results and lowered the standard deviation. However, the participants were not sure whether the statistics improved their results. The participants´ and the moderator´s experience with the method enhance confidence in the results.

Data analyzed using Grounded Theory: Here, we use the answers from questionnaire 5: One week after the experiment, during the post-test session, each participant received his/ her priorities resulting from each method. They were asked to comment whether the results reflect their opinion, on the methods and why results of different participants deviate so much from each other.

The *second experiment* was a pilot experiment to prepare the third experiment. Objective: Experiment 2 and 3 both tested the reliability, i.e. repeatability of risk estimations. Experiment 2 focused on the qualitative results, to learn which factors might influence repeatability and how to design Experiment 3.

Sample population**:** The participants were the author of this study and three students of the University of Braunschweig in the winter term 2009/10. The students had not learned about risk-based requirements prioritization before.

System: The experiment reused two case studies from former experiments: Experiment 1 and a second case study, a ticket vending machine. The latter experiment considered three risks and six requirements.

Material: The participants got the introduction from Experiment 1 describing the case study, while the ticket vending machine was illustrated by screenshots. The risk estimation questionnaires from previous runs of the same experiment were reused.

Execution: After a 20 minutes introductory presentation about the methods and case studies, both experiments were executed once. Then, each of the participants executed a second and third run at home, after at least two days. The author of this publication executed each estimation five times. Assumptions made were noted.

Results: The probability and damage estimations varied strongly. Two participants better repeated their risk estimations, while the other two produced risk estimations which seemed to stem from different persons.

Data analyzed using Grounded Theory: The list of assumptions made during the estimations and the results of the joint discussion of why repeatability might be low.

The *third experiment* repeated Experiment 2 with more persons and improved material.

Objective: The experiment investigated the repeatability of risk estimations and which factors influence the repeatability of risk estimation.

Sample population**:** Eleven bachelor and master students of the University of Heidelberg in the winter term 2010/11. They had been taught prioritization methods and risk-based requirements prioritization in a lecture before, and applied four different prioritization methods in homework. Methods and case study used in the experiment, however, were new.

System: The system used as a case study was the e-learning system which the students know and use for two to eight semesters already.

Material: The material included the case study description with screenshots and explanations, which partly quantify the risk damage, e.g. in terms of hours of downtime, and two questionnaires supporting one prioritization method each – the risk-based requirements prioritization and ranking.

Execution: The first run of the estimations took place in a lecture, the second as homework, the third and fourth in the lecture a month later. Each run took 30 minutes.

Results: The risk estimations have been found to be badly repeatable again. The quantitative results will be published elsewhere.

Data analyzed using Grounded Theory: Each of the prioritization questionnaires asked for assumptions made and for information which might have been missing. Further questions: Are there requirements or Misuse Cases where you have been especially uncertain? Which and why? Which information was missing? Do you think that risk estimation leads to realistic and practically useful results? Why or why not?

4 Grounded Theory

The Grounded Theory was developed by Strauss and Corbin [StCo90] for building theories from qualitative data (like protocols from interviews, research publications and other text of any form) in social research. It structures the data in a traceable way. We have successfully applied this method for analyzing results from empirical software engineering research and literature research before (e.g. in [RDH10]). The steps of Grounded Theory are [StCo90]:

- *Asking the research question*: “The research question [...] is a statement that identifies the phenomenon to be studied. It tells you what you specifically want to focus on and what you want to know about this subject.” ([StCo90], p.38)

- *Reading technical and non-technical literature to stimulate theoretical sensitivity*: “Theoretical sensitivity refers to the attribute of having insight, the ability to give meaning to data, the capacity to understand, and capability to separate the pertinent from that which isn´t.” ([StCo90], p.41f) This sensitivity can be reached by reading, by practical work or during the research.
- *Open coding*: Data are analyzed in order to identify concepts, grouping them to categories, identifying the categories´ properties and dimensions.
- *Questioning for enhancing theoretical sensitivity*: The categories are discussed, especially with respect to the data.
- *Axial coding*: Data are structured by making connections between categories; this is done by utilizing a coding paradigm involving causal conditions, phenomena, context, intervening conditions, action/ interaction, consequences.
- *Selective coding*: This means the identification of the core category, to write the story and story line.
- *Identification of process* means linking of action/ interaction sequences and contingency (unplanned happening).
- *Transactional analysis by a conditional matrix*: “The conditional matrix may be represented as a set of circles, one inside the other, each (level) corresponding to different aspects of the world around us. In the outer ring stand those conditional features most distant to action/ interaction; while the inner rings pertain to those conditional features bearing most closely upon an action/ interaction sequence.” ([StCo90], p.161)
- *Theoretical sampling*: The empirical data are sampled, using the concepts and categories found before. The objective is the discovery of as many relevant categories as possible, along with their properties and dimensions.

These steps can be and usually are traversed iteratively several times.

5 Results

This section summarizes the results of the Grounded Theory analysis of the qualitative results (text answers from participants on the questionnaires) of the three experiments described in Section 3. Due to lack of space, we cannot trace each step of the analysis and not present the results in their chronological order of emergence. This chronological order would describe a consecutive structuring of the data in a bottom-up fashion: From the raw data, abstracting properties were identified and grouped into categories, then linked to each other in “condition -> action -> consequence” stories. Finally, the core category is identified and all other results grouped around it. In what follows, the analysis results are summarized top-down instead.

The research question is: “What information is needed for IT risk estimation?”

Consistent with the research question and topic of the experiments, the *core concept* was chosen to be *risk*, which describe a deviation from the specified system use. The *phenomenon* to study are Misuse Cases.

It was found that the axial coding concepts directly map to the Misuse Case concepts: The *causal condition* of a Misuse Case is a user who tries to execute a functionality and/or a misuser who is motivated to execute a threat, what is an action which deviates from the specified use. Intervening conditions are vulnerabilities (which enhance the probability of the Misuse Case or the damage caused), user and misuser properties. Context is characterized by the point of time when misuse happens and when a countermeasure is applied, and by the properties of the reference system against which the risk estimation is performed (its functionalities, technical implementation and countermeasures taken). Action/ interaction are described by the Misuse Case scenario. Consequences of the Misuse Case can either be success (i.e. specified use) or damage, because the Misuse Case by definition happens with a certain probability and therefore damage might occur or not. In what follows, the Misuse Case concepts are used as the categories to structure the data.

The story line emerging from the data is presented graphically in Fig. 1 in the form of an event tree. It is important to mention that this is not the usual model used to describe Misuse Cases, but it is more detailed. Misuse Cases can follow each other, i.e. the “damage” of one Misuse Case can be the threat which initiates another Misuse Case. The experiment participants asked a lot of what-if questions about probabilities of different event flows, like “How many users use the countermeasure?” or “What happens if the countermeasure is not effective?” or “When the Misuse Case happens, does this mean that the student is excluded from the exam?” (Remark: Success and damage are defined from the user perspective, not the misuse perspective. A successful hack of a database by hacker therefore is counted as damage.) Note that the risk probability estimated in the experiment should correspond to the sum of the probabilities of all scenarios which end with “damage”. However, as these numerous different possible flows of events were not specified in the experiment material, it is possible that the estimators were not aware of this complexity, and this can be one reason for the deviations between risk estimations of different participants and for the participants´ feeling that risk estimation is rather difficult to execute and the results might partly not be too realistic.

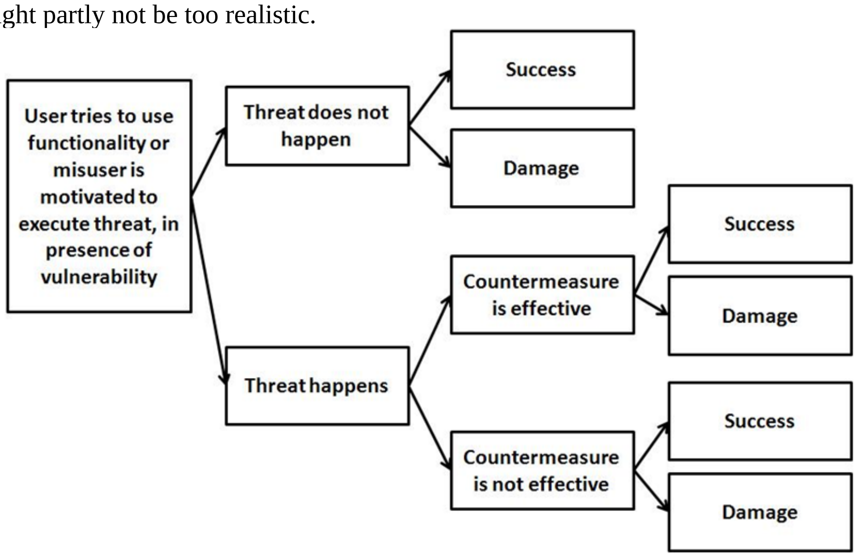


Fig. 1. Story line.

Open coding was a bottom-up analysis of the data. We present separately the information about reality (i.e. system and system (mis)use), and the information which is specifically needed for risk estimation. Both information types together are needed for IT risk estimation. **Table 1** summarize the results of open coding which are categories (i.e. concepts), properties of the categories and dimensions of the properties. The tables also give each concept´s category of axial coding and its category from the conditional matrix. When doing a simple transactional analysis, only few categories were found: There is the system, its users, the organization where the system is operated (company or university) and the world outside this organization. For classifying the present data, no more categories were needed.

Table 1. Analysis results by open coding, axial coding and conditional matrix: What information describes the system and system (mis)use?

category/ concept	property	dimensions	axial category	conditional matrix category
reference system	countermeasures	further countermeasures implemented	context	system
	functionalities	tasks, steps, importance, number, frequency, support by system, number of screens	context	system
	technical properties	implementation of database, hardware availability, further vulnerabilities	context	system
countermeasures	degree of usage by users	sometimes, often not	intervening conditions	users
	effectiveness against a certain risk	how often, to what degree? sometimes, considerably	intervening conditions	system and users
	alternatives to countermeasure	available or not; which: workaround, other media/ system functionality for achieving the same objective (e.g. e-mail submission of homework, instead of using the e-learning system, the browser´s back button)	intervening conditions	system and users
	risks of the countermeasure which make it less effective	damage happens nevertheless, technology failure, user error	actions	system and users
	countermeasure implementation	easily found? Understandable?	intervening conditions	system
points of time	point of time when countermeasure is applied	before submission of homework, after submission of homework	context	users
	point of time when Misuse Case happens	a day before homework deadline, directly before deadline	context	users
consequences	consequences of consequence	consequences for higher goals like admission to exam, pattern (how many data are concerned, does it happen regularly?), alternative ways for reaching a goal exist	consequences	system and users
users	user role	student, study domain, developer	intervening conditions	users
	technology usage skills	Moodle, internet browsers, web interfaces	intervening conditions	users
	general knowledge	able to read, intelligent	intervening conditions	users
	do they use countermeasure?	sometimes, often not, are angry when must use countermeasure	intervening conditions	users
	behaviour in case of misuse case	accept deficit, migrate to competitor, achieve goal by trial and error, organizational countermeasures taken, do they hear of security problems?	intervening conditions	users
	experience in using this system	for the first time	intervening conditions	users
misusers	motivation	how many want to execute the misuse case?, effort put into executing the misuse case, misuser´s abilities	intervening conditions	users and rest of the world
system environmen	organization within which system is operated	University Heidelberg	context	organization
	technological context	interfaces to other data systems	context	organization
	market	properties of competing systems	context	rest of the world

Table 2. Analysis results by open coding, axial coding and the conditional matrix: What information is needed specifically for risk estimation?

category	property	dimension
probability	metric	number of times, precent
	transformation of scales	How many times correspond to 1%?
	statistics	historical data
damage	metric	image, cost
perspective of estimator	role	user, maintainer, developer
	consistent with perspective of misuse case?	yes, no
	experience	completeness: some misuse cases are not experienced, although they happen
	priorities	according to user perspective, relative priorities of quality attributes (e.g. security versus usability)
	experience and knowledge	technical, market
context of estimation	time pressure	yes
system goals	task	submission of homework
	constraint on tasks	e-learning: each student does his homework without knowing the solutions of others
	relevance of data	relevant for lecture or not

6 Discussion of Validity

Validity of empirical research means that the results reliably answer the research question posed and that it is known for which conditions these results are valid and where they are not. The research question investigated here is: “What information is needed for IT risk estimation?”

The data used for answering this question stem from three experiments where IT risk was estimated. These experiments investigated other research questions than the one of this publication, the data were gathered asking questions which were different in each of the experiments. These questions mainly asked for missing information and for difficulties in using the method. This means we use material from heterogeneous experiments and questions. There might be a threat for validity due to this, however, only a small one as in all these questions the experiment participants mainly discussed missing information, even when not being explicitly asked for that. We expect that the information that was given in the experiment material is not mentioned in these answers. However, this given information can also be described by the model in **Table 1** and **Table 2**, i.e. it would not add further categories.

The categories found seem to be saturated. For most of the categories, several instances were found in the data, often five to ten. We analyzed more than 100 sentences or keywords, and after half of them, only very few new categories emerged. However, the dimensions evidently are not saturated. For instance, for the property “degree of usage of the countermeasure”, only two dimensions were found (“sometimes” and “often not”), although there exist more possibilities like “always” or “never”.

One reason for the low number of dimensions is that often the data had the form of questions, like: “Do users apply this countermeasure?” instead of “I assume that users often do not apply this countermeasure”. Such questions contributed to the model categories and properties, but no dimensions. Either more data than that of 46 questionnaires resp. three experiments would be needed for completing the dimensions, or questions should be asked which rather ask for assumptions than missing information. Whether the properties are saturated, we cannot say. The low number of three systems analyzed also poses a threat to validity. It is possible that with respect to other systems, further categories, properties and dimensions would have emerged. These partly are system-specific.

In Grounded Theory, usually it is recommended that more than one scientist does the coding of the data. In the present research, this was not possible, as it was executed by one person only. Therefore, subjective classification errors would not have been discovered.

Open coding demands much discipline to keep to the data (and not add categories to the model that is not based on data), especially when categories start to emerge which one knows from a long-used conceptual model like the Misuse Cases in this study. However, in our Grounded Theory protocol, the original data are traceably mapped to the categories, to make sure that no concepts enter the resulting model without being supported by real data and vice versa, that all data fit into these categories and no data stay unclassified.

The experiment participants as well as the coder know Misuse Cases and their concepts. This easily explains why the Misuse Case concepts were found in the data and these are overrepresented against concepts which are not part of a Misuse Case description (like “point of time”). It is possible that if the involved persons did not know Misuse Cases, different or further concepts would have been found.

We can assume that IT risk estimation in practice and IT risk estimation in an experiment is a different situation. For instance, we have observed in an unpublished (confidential) case study where we applied IT risk estimation in a real software project that practitioners who are experts of the system feel much more confident about their IT risk estimations and in the using risk-based requirements prioritization method, compared to students in an experiment. The practitioners know the system to treat very well, including the developer perspective. And they may modify the wording and granularity of Misuse Cases and requirements and even modify the method according to their need. Experiment participants may not modify method, Misuse Case descriptions or requirements, and have no possibly to research for statistics. This turns risks estimation more difficult in an experiment using hypothetic scenarios or even a fictitious system. In the experiments, therefore, we have always chosen a system which the estimators know at least from the user perspective by regular use. However, they were aware that their own experience with the system is not representative. While these factors reduce the ease of IT risk estimations and its quality, they are rather helpful in investigating which information is needed for IT risk estimations. We expect that a student experiment can provide for a more complete model of information than asking practitioners, who use implicit expert knowledge without being aware of it.

7 Summary and Conclusion

This publication presents the results of a Grounded Theory analysis of qualitative data from three experiments about risk estimation, i.e. from 46 questionnaires. This analysis follows the research question "What information is needed for IT risk estimation?" Knowing what information is needed helps to prepare a risk estimation workshop, in real-life workshops as well as in teaching and scientific experiments.

Our findings imply that risk estimation needs diverse information about the system and its context, and good descriptions of the risk scenarios – the success scenarios as well as the diverse flows of the damage scenarios. These scenarios could be described as event trees, better than with the conventional Misuse Case descriptions. Based on our findings from our experiments, we recommend to provide risk estimators with the following material:

- A case study description including the success scenarios of system use, the technology and users, and the context. More details about what is needed can be found in **Table 1** and **Table 2**. As this is a lot of information, instead of defining all this a priori, one can discuss and define the assumptions jointly during the estimation workshop.
- Descriptions of the success and risk scenarios in the form of event trees, where the risk of each step and branch should be estimated separately. As before, we would illustrate these scenarios by adding screenshots of the system for different scenarios (although our experiment results say nothing about the usefulness of the screenshots). Quantification seemed helpful, like defining the number of hours of a system breakdown.

Due to validity threats to the results (which are stronger for the dimensions than for the categories), the model developed in this publication probably is not yet complete. Furthermore, as we have discussed above, that both experiment participants and researcher know Misuse Cases well may have influenced the data in the experiments and the model based on it. Therefore, in future it would be interesting to gather similar data by executing further experiments with participants who do not know Misuse Cases. Nevertheless, they should have a computer science background or at least substantial software user experience in order to discuss about IT risks competently.

References

[AHP04] Arora, A., Hall, D., Pinto, C.A., Ramsey, D. and Telang, R.: An ounce of prevention vs. a pound of cure: How can we measure the value of IT security solutions? Carnegie Mellon CyLab. (2004)

[BF00] Boehm, B.W., Fairley, R.E.: Software Estimation Perspectives. IEEE Software 17(6), 22-26 (2000)

[BSI05] BSI: BSI-Standard 100-1: Managementsysteme für Informationssicherheit (ISMS) Version 1.0, https://www.bsi.bund.de/cae/servlet/contentblob/471450/publicationFile/30749/standard_1001_pdf.pdf (2005)

[BF04] Bundschuh, M., Fabry, A.: Aufwandschätzung von IT-Projekten. 2. Edition, mitp Verlag, Bonn (2004)

[CER06] CERT/CC Statistics 1988-2006, http://www.cert.org/stats/cert_stats.html

[Dav03] Davis, A.M.: The Art of Requirements Triage. IEEE Computer 36(3), 42—49 (2003)

[FC03] Feather, M.S., Cornford, S.L.: Quantitative Risk-based Requirements Reasoning. Requirements Engineering Journal 8(4), 248—265 (2003)

[FCG01] Feather, M.S., Cornford, S.L., Gibbel, M.: Scalable Mechanisms for Requirements Interaction Management. In: IEEE International Conference on Requirements Engineering, Schaumburg, USA, 119—129 (2001)

[FCK06] Feather, M.S., Cornford, S.L., Kiper, J.D., and Menzies, T.: Experiences using Visualization Techniques to Present Requirements, Risks to Them, and Options for Risk Mitigation. In: Int. Workshop on Requirements Eng. Visualization, Minneapolis/ St. Paul, Minnesota (2006)

[FCL00] Feather, M.S., Cornford, S.L., Larson, T.: Combining the Best Attributes of Qualitative and Quantitative Risk Management Tool Support. In: 15th IEEE International Conference on Automated Software Engineering, Grenoble, pp. 309–312 (2000)

[Her10] Herrmann, A.: Praktische Tipps für die Risikoschätzung und risikobasierte Anforderungspriorisierung. iqnite, Düsseldorf, Germany (2010)

[HP08] Herrmann, A., Paech, B.: MOQARE: Misuse-oriented Quality Requirements Engineering. Requirements Engineering Journal 13(1), 73—86 (2008)

[HP09] Herrmann, A., Paech, B.: Practical Challenges of Requirements Prioritization Based on Risk Estimation. Journal of Empirical Software Engineering 14(6), 644--674 (2009)

[How68] Howard, R.A.: The foundations of decision analysis. IEEE Trans. Systems, Sci. Cybernetics 4(3), 211–-219 (1968)

[ISO02] ISO (International Standards Organization): ISO, Risk management – Vocabulary – Guidelines for use in standards, ISO Guide 73. Geneva: International Standards Organization (2002)

[PPB99] Park, J., Port, D., Boehm, B., In, H.: Supporting Distributed Collaborative Prioritization for WinWin Requirements Capture and Negotiations. In: Int. 3rd World Multiconference on Systemics, Cybernetics and Informatics SCI´99, Vol.2, 578--584 (1999)

[RDH10] Racheva, Z., Daneva, M., Herrmann, A.: A Conceptual Model of Client-driven Agile Requirements Prioritization: Results of a Case Study. ESEM Conference 2010, Bolzano (Italy)

[RRM02] Raiffa, H., Richardson, J., Metcalfe, D.: Negotiation analysis - the science and art of collaborative decision making. Cambridge: Belknap (2002)

[Ric03] Richardson, R.: CSI/FBI Computer Crime and Security Survey, Computer Security Institute, http://i.cmpnet.com/gocsi/db_area/pdfs/fbi/FBI2003.pdf (2003)

[REP03] Ruhe, G., Eberlein, A., Pfahl, D.: Trade-Off Analysis For Requirements Selection. Int. J. Software Eng. And Knowledge Eng. 13(4), 345--366 (2003)

[SO00] Sindre, G., Opdahl, A.L.: Eliciting Security Requirements by Misuse Cases. In: TOOLS Pacific 2000, 120--131 (2000)

[SO01] Sindre, G., Opdahl, A.L.: Templates for Misuse Case Description. In: 7th Int. Workshop on Requirements Eng.: Foundation of Software Quality – REFSQ, Essener Informatik Beiträge Band 6. Essen, Germany, pp. 125--136 (2001)

[Sta03] Stamatis, D.H.: Failure Mode and Effect Analysis - FMEA from Theory to Execution. Milwauki, USA: American Society for Quality Press (2003)

[StCo90] Strauss, A.L., Corbin, J.M.: Basics of qualitative research - grounded theory procedures and techniques. 6th print, Sage, Newbury Park, USA (1990)

[TK74] Tversky, A., Kahneman, D.: Judgment under uncertainty: Heuristics and biases. Science 185, 1124—1131 (1974)

Defining Product Lines and Product Variants based on Prioritization of Customer Segments and Customer Requirements

Andreas Helferich[1], Lars Oliver Mautsch[2]

[1] Andreas Helferich – Software Management Consultant,
Braystr. 18, 81677 Munich, Germany
mail@andreashelferich.de
[2] Universität Stuttgart, Chair of Information Systems II (Business Software),
Keplerstr. 17, 70174 Stuttgart, Germany
mautsch@wi.uni-stuttgart.de

Abstract. Quality Function Deployment (QFD) is a well-known Quality Management and Product Engineering method used in various industries all over the world. It can also be used in Software Development, where it is usually used to prioritize requirements. We extended QFD in a way that allows for the prioritization of requirements and also customer segments, which in turn allows for the definition of (Software) Product Lines and the Product Variants included in a Product Line. Our approach – called Quality Function Deployment – Product Portfolio Planning (QFD-PPP) enables software companies to base decisions on core and variable assets on actual customer needs and therefore assists in controlling product (and especially architectural) complexity.
Besides explaining the method and related approaches, this paper also describes the successful use of QFD-PPP in an industrial case study in which a standard software package used in data center operations was analyzed with regard to possible variants.
Keywords: Case Studies, Product Portfolio Planning, Product Variants, QFD, Software Product Lines

1 Introduction

Serving the needs of a large number of customers with different needs is one of the distinguishing features of commercial off-the-shelf (COTS) software packages. This is achieved through including a large number of features and a significant degree of built-in flexibility, allowing the customers to adapt the software to their specific needs. This allows software companies to “mass-produce” COTS software packages (or rather to develop the software once, make copies of the software with hardly any effort and sell it to a potentially large number of customers), resulting in significantly lower prices compared to custom-developed software. Where different customers’ requirements are relatively similar but diverse when analyzed in further detail, offering a number of distinct variants of the software package can be beneficial.

A large number of software development techniques have been developed to cope with this complexity. Still, every variant developed results in additional effort.1 Research on complexity management shows that purposefully controlling complexity by limiting the number of variants offered can be very beneficial. The proponents of Software Product Lines take this into account by arguing that an activity called Scoping should precede the main development effort. Scoping is the process during which decisions about what to develop, i.e. which products will be part of the product line and what the commonalities and variabilities will be, are made. At the same time – and maybe even more important – it is also determined what not to develop, i.e. the product line is bound on several levels building upon each other [1]: Product Portfolio Scoping, and Asset Scoping.

Currently, no single scoping approach addresses all three levels. One of the most comprehensive scoping approaches is the PuLSE-Eco approach [1]. Unfortunately, current approaches to Product Line Scoping usually treat the number of variants to be offered and the customer segments targeted with each variant as input from Marketing [1]. We designed QFD-PPP to support decisions on the product portfolio and to limit the number of variants offered according to actual customer requirements.

As pointed out previously [2], QFD-PPP allows software companies to develop a platform-based product portfolio (possibly implemented using Product Line Engineering) that optimally satisfies customer demands and at the same time restricts the number of products offered. Focus of this paper is to demonstrate how QFD-PPP can support software companies' portfolio planning. To this end, we begin by briefly describing QFD, Software Product Line Engineering as well as existing variations of QFD used to define product variants. Following this, we describe QFD-PPP and present the results of a case study performed with a German software company offering standard software used in data center operations.

2 Quality Function Deployment

2.1 (Software) Quality Function Deployment

QFD has been developed in the Japanese manufacturing industry [3], [4] and has been successfully applied to develop customer-oriented products in services in a variety of industries [5]. Compared to traditional ways of formalizing and specifying product requirements, "QFD provides a systematic but more informal way of communication between customers and developers". [6] Instead of trying to compile an exhaustive list of requirements that specifies the requirements in great detail, QFD aims to identify the most important requirements and to define a vision for the product to be developed. Therefore, a project team consisting of customer representatives, developers/engineers and QFD-experts serving as facilitators collaborates during the whole QFD process. The mission of this team is to assure that the final product's features are not determined by the technically possible but by the fitness for use,

[1] Actually even variants not yet developed but intended to be offered in the future and already considered in the software architecture add a certain level of complexity.

i.e. the features the customers demand. The software developers and/or engineers assure that the features can be implemented and that technological breakthrough innovations are not ignored.

In order for QFD to be used for software development, two differences are considered: first, the software production process is basically a duplication process and implementation is largely determined by the system design, especially the system architecture. Therefore, the effort has to be directed mainly into the earlier stages. Secondly, "Software […] is valued not for what it is, but for what it does" [7]. Thus, the distinction between product functions and quality elements has to be made: a product function is a "functional characteristic feature of the product, usually not measurable (creates perceptible output)" [8], while a Quality Element is a "Non-functional characteristic feature of the product, possibly measurable during development and before delivery (does not create perceptible output)" [8].

The most important purpose of QFD in software engineering and the main focus of product planning is on setting prioritized development goals based on the most important customer requirements [8].

The best-known instrument of QFD is the so-called House of Quality (HoQ) [9]. Generally speaking, the HoQ is the matrix[2] which analyzes customer requirements in detail and translates them into the developers' language. Since QFD is a very flexible method, a QFD project can use a number of matrices or none at all. Still, the HoQ and related matrices are at the core of most QFD projects. Frequently, these matrices are linked with each other with the output of one matrix serving as the input of another matrix. Various models describing possible deployments and matrix sequences exist, applicable for software development are (among others) the deployment framework developed by the German QFD-Institute [10], Zultner's Comprehensive Software Quality Deployment [11] or Ohmori's Software Quality Deployment [12]. For an in-depth description of QFD and its application in various contexts see [3], [4].

But applying QFD is more than filling out a HoQ matrix. A number of techniques (e. g. Conjoint Analysis [13], Target Costing [14] or New Lanchester Theory [15]) can be used so as to provide the most benefit.

In order to make sure all relevant points of view and all available knowledge about customer requirements are included in the process, the entire QFD process is carried out by a project team consisting of representatives of all departments (development, quality management, marketing, sales, service etc.) and is to be extended in several team meetings by carefully selected customer representatives. Substituting a customer survey, one of the first meetings tries to ascertain customer needs and to classify them in the so-called Voice of the Customer (VoC) Table. These requirements are structured using affinity- and tree diagrams and weighted (e. g. by pair-wise comparison or the Analytic Hierarchy Process [16]) by as many members of the customer groups as possible under control of the customer representatives. The weights of the different groups are then used to calculate the average weight by calculating the average of the weights assigned by the customer groups weighted with the importance of the groups.

[2] In QFD, a matrix is basically a table contrasting two factors and their correlations, for the HoQ these are Customer Requirements and Technical Requirements.

If a new release of an existing product is developed, customer requirements can be used to evaluate the level of satisfaction with the current fulfilment of the requirements (measured on a scale ranging from 1 indicating total dissatisfaction to 5 indicating perfect satisfaction). If sufficient and reliable information about competing products is available, these can also be evaluated using the customer requirements collected. Analysis of the satisfaction with the competing products supports decisions regarding the focus of the new release. The second major input is the Voice of the Engineer Table, compiled by the QFD team, among them particularly developers, that includes potential product functions and measurable quality elements. Both are derived from the requirements by the developers. The relationships between product functions and customer requirements are identified together with the customer representatives. Figure 1 displays an excerpt of a Software-HoQ for an email client including the tables of customer requirements and product functions.

Customer Requirements / Product Functions	Weight in %	Enter email via voice	Spell and grammar check	Create personal addres book	Filter incoming emails accoring to criteria	Reject emails from certain users or domains	...
Write emails fast/easily	7.2	9	3	3			
Write emails fast to many users	5.3	9	3	9			
Have overview of incoming emails	8.1				9	3	
Write emails not using your hands	6.4	9					
Emails grammatically and orthographically correct	2.3	3	9				
...	...						
Difficulty level		9	3	3	1	1	...
Competitor A (better / worse)							...
relative Importance		26%	16%	34%	16%	8%	...
absolute Importance		450	270	585	270	135	...
Ranking		2	3	1	3	5	...

Fig. 1. Excerpt of the Software-HoQ for an email client [8].

2.2 Software Product Lines and Product Line Engineering

While the Marketing definition of a Product Line does not make any implications whether the members of the product line are technically related [17], the term “Software Product Line” implies that different products of one domain (also referred to as problem space or application range, e. g. operating systems for mobile telephones or software support of the sales department) are viewed as a product family and not as single products. According to the Software Engineering Institute, Software Product Lines are defined as “set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way“. [18] The components of a Software Product Line are the product line architecture and the individual products which are part of the product line. The product line architecture describes the individual products, their common components and - at least in outlines - the differences between the products of the family (e.g. [18]). Different process models exist for the development process of product lines, e. g. those described in [19] or [20]. They have in common, that the product line development process is modeled along the structure of a product line. Just as the product line consists of product line architecture and product line members, the development process also consists of the process of the development of the product line architecture and the development process of product line members. The development of the product line architecture is called domain engineering and the development of product line members is called application engineering.

As briefly mentioned in the introduction, preceding both is an activity called scoping during which the use of the product line or its products is planned (e.g. [1]). More explicitly, scoping aims at distinguishing between requirements that are outside of the scope of the product line and requirements that are within the scope of the product line. The latter are further distinguished in requirements common to all products and variable requirements, i.e. requirements that only one or some of the members of the product line need to fulfill. For a further description of the activities and results of Software Product Line Engineering see for example [18].

3 Quality Function Deployment-Product Portfolio Planning (QFD-PPP)

There are a few examples in literature where QFD was used to define product variants. In [21], these are analyzed and an explanation is given why these examples fall short of realizing the full potential of applying QFD for Software Product Lines. To fix these shortcomings, we developed QFD-PPP.

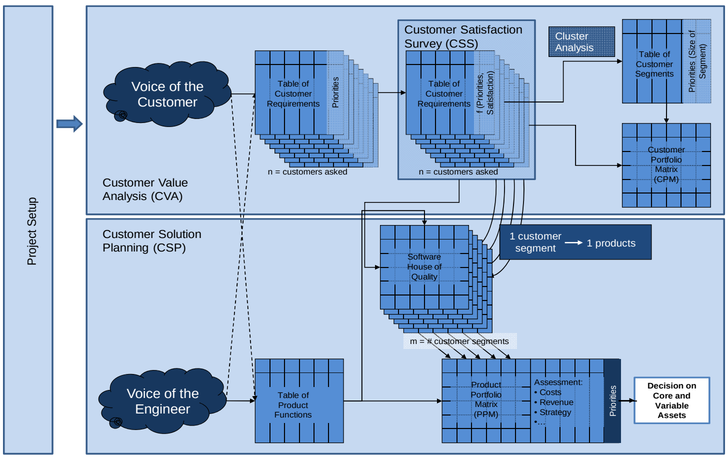


Fig. 2. Overview of QFD-PPP (simplified).

Our approach to Product Portfolio Planning makes extensive use of QFD while at the same time introducing two new matrices (see Figure 2). First of all, the Voice of the Customer (VoC) is collected by asking existing and potential customers about the requirements they have for the product line. Once these answers are collected, they are analyzed and sorted before asking the customers to assign priorities to all requirements. Once these priorities are assigned, customer segments are formed based on these priorities using cluster analysis. Thus, unlike traditional QFD approaches, there is no weighting of customer groups as this is only necessary to come up with common (or rather average) priorities. Another difference is the identification of customer groups not by attributes of the customer (e.g. job title or role description) but by statistical analysis. In this regard, our approach also differs from approaches which identify customer segments and their sizes based on the products purchased and their market shares (e.g. [15]).

The next step is to get together developers, software architects and selected customers (based on the clusters identified) to build the Software House of Quality. Explicitly including the Voice of the Engineer in the form of product functions is important to identify innovative software characteristics that customers themselves would not have come up with.

Since a product function's level of fulfilling a customer requirement is independent from the weight assigned to the requirement, there is basically only one Software-HoQ for all the members of one product line. But since the weights of the customer requirements depend on the customer segments, the weight of the product functions does so either. Therefore, there is a different Software-HoQ for each product variant, but the different Software-HoQs differ only in the weights of customer requirements and product functions. The Software-HoQ in Figure 1 equals the Software-HoQ for one of the customer groups (including the weights) for an email-client, e.g. attorneys used to dictate letters who would therefore being able to dictate emails, too. The first new matrix is called **Customer Portfolio Matrix**. It provides an overview of all customer requirements and customer segments, including the importance assigned to the requirements, and is displayed in Figure 3.

Customer Requirements / Customer Segments /Products (in %)	Product Line Member/ Customer Segment #1	Product Line Member/ Customer Segment #2	Product Line Member/ Customer Segment #3	...
Write emails fast/easily	7.2	5.3	6.8	
Write emails fast to many users	5.3	6.3	7.5	
Have overview of incoming emails	8.1	11.2	8.1	
Write emails not using your hands	6.4			
Emails grammatically and orthographically correct	2.3	7.4	5.3	
...				

Legend
- : 10% most important customer requirements
- : 25% most important customer requirements

Fig. 3. Customer Portfolio Matrix.

As a working hypothesis, for each customer segment exactly one product variant is designed (as indicated in Figure 2). Core and variable features are identified by comparing the weight of the product functions for the different customer segments. This is visualized in the second new matrix called Product Portfolio Matrix that relates product functions to members of the product line. This matrix is not a priorization matrix such as the HoQ, but rather a way of displaying relevant information next to each other in order to facilitate decision-making. For example, the software company could decide to offer a very advanced spell and grammar check as part of all product line members (as displayed in Figure 4) order to a) provide an advantage with regard to competitors A and B and b) avoid the complexity that would result from offering different spell checks.

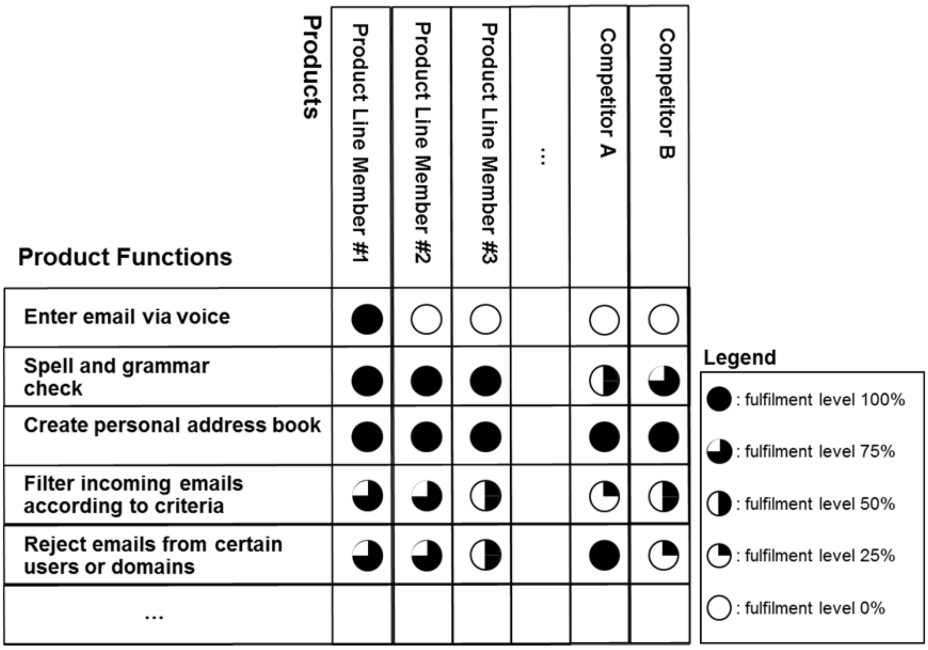


Fig. 4. Product Portfolio Matrix (excerpt).

Thus, the Product Portfolio Matrix relates product functions and members of the product line and helps prioritizing the variants. Additional inputs are the expected costs for the product functions and the expected revenue a product will achieve. The second depends on the size of the potential market, the products currently available on the market and the customer satisfaction with these products and the advantage the member of the product line have over these products. Ulwick's so-called *opportunity algorithm* [22] or the algorithm used in [8] can be used as indicators here. Both algorithms use the importance of a feature and the customers' satisfaction with the current solutions provided by own and competitors' products to identify features where improvements provide a competitive advantage. A more detailed economic assessment is presented in [1].

In addition to the described priorization and decision-support for functional requirements and characteristics, priorization and decision-support for non-functional characteristics and software quality attributes are enabled. To this end, the software developers and software architects evaluate different software architectures and technologies taking into account necessary quality attributes. This is also done by using matrices, where the roof is intensively used to analyze the impact that different architectural or technological elements have on each other. The results of this analysis are used to decide on the software architecture and the technologies to be used for prototypes. If necessary, an in-depth evaluation of architecture alternatives using methods such as SACAM or ATAM can be executed [23].

These prototypes can also be used to demonstrate exciting features the software developers and software architects came up with and the proposed solutions to the requirements voiced by the customers. If all prototypes are shown to all customers, some of the customers will decide to include some features they previously hadn't assigned value to, maybe drop some features they requested. This discussion is based on the product functions, not the original customer requirements and their weights. Only when large changes are asked for, the customer requirements will be re-evaluated.

Additionally, derivation of new products for a Software Product Line and the evolution of the Software Product Lines and its members are facilitated, since the already existing matrices can be used as templates (a similar course of action for agile software development was proposed in [24]). Using the matrices as a starting point leads to reductions in both time-to-market and costs and helps achieving important goals associated with Software Product Lines.

4 Case Study: Job Scheduling Software

This case study was conducted in cooperation with a German software company that currently offers a standard software package in the field of datacenter operations. Due to technical reasons, the company intended to completely redesign the software. At the same time, they were considering to serve another market segment using a variant of their product with reduced functionality compared to the main product. As a long-term option, they were also interested in offering every customer a version of the product that was tailored to his needs. The current product is offering extensive functionality for a comparatively modest price, whereas management had decided the product variant was to be positioned as easy to use and quite competitively priced. The company approached us in order to a) assure that the resulting small product line met customer requirements and b) analyze the differences in customer requirements between the projected customer segments for the two members of the product line.

During project setup, this objective was defined in more detail: the present release plan of the standard software should be evaluated (and if necessary adapted) with regard to the coverage of the requirements of the currently served market segment. Additionally, competing products should be examined referring to these requirements and their fulfillment. Furthermore, we were asked to investigate how far the requirements of the targeted new market segment differ from those of the currently served market segment and which consequences these differences have on the product characteristics.

After the objectives were defined, a first workshop was conducted to collect and structure the customer requirements. Therefore representatives from marketing and sales as well as developers from the German software company, representatives of several customers and representatives of a potential customer as well as an industry expert were part of the project team. An external marketing expert, who was responsible for developing a new marketing concept for the software package, was also part of the team. The end users of the software, the so-called operators, and their supervisors were identified as relevant customer groups.

The responsibility of the operators is to assure that different software programs are executed in their data centers according to plan. The supervisors need to know about any exceptions and problems, since they are responsible for quality of service and will receive complaints from the data centers' customers. Representatives of both groups were invited to and participated in the process. The Voice of the Customer Analysis generated 71 customer requirements, which were divided into 12 requirement categories. Then the requirements were weighted by the customer representatives. Figure 5 displays an excerpt of the customer requirements including the results of a customer satisfaction survey and a comparison with two competing products that were also conducted with the participants of the workshop.

Members of the additional customer segment that was intended to be targeted with the "scaled-down" variant of the software were included after this workshop. In order to get an even more detailed impression of their requirements, in-depth interviews with three companies were conducted. During these interviews, the customer requirements identified during the workshop were discussed. According to the characteristics given by marketing, all three companies belonged to the market segment targeted with the scaled-down version. The objective of this discussion was to evaluate a) in what way the potential second market segment differs from the currently served customer segment and b) if this second market segment is actually homogeneous enough to be targeted with one single product variant. As expected there were some common patterns of difference in the importance of the customer requirements: rather basic requirements were relatively rated higher, whereas a few of the advanced requirements were rated lower.

Looking at the results for the new market segment in further detail, there were unexpected differences between the requirements of the three companies. E.g. one of the interviewed companies would need this software primary for the support of its in-house SAP system, so that several requirements such as multi-client capability or business process monitoring wouldn`t be relevant. The second company instead wouldn`t need the support of different time zones, as it currently operates just in Germany, unlike the other two companies. The third company focuses especially on the multi-client capability as it operates as a service provider offering the administration of IT systems for other companies. These differences lead to the conclusion that the targeted market segment was not as homogeneous as expected. While it was clearly defined (companies with a midsized datacenter, which usually did not use this kind a software so far, but had the operators fulfill this task manually or using simple scripts), pretty large and not served well by existing products, the customer needs within the segment are too inhomogeneous to be covered by one single clearly defined product (respectively such a product would only be purchased by parts of the segment). This reflects a well-known problem in marketing:

any segmentation approach should lead to segments that are easy to define and target with your marketing and sales campaigns and at the same time highly relevant in predicting purchasing behavior [25]. Obviously, the segment planned here was easily targeted but not very relevant in predicting purchasing behavior since customer needs vary too much. As a result the German software company decided to implement a modular design which consists of a basic system and several additional modules which are independent from each other. This way, the planned additional market can be served with the pretty aggressively priced basic system, making it attractive to first-time users. At the same time, the company can demonstrate to the currently served market segment how powerful the “all in”-system is and how long reference customers have been using such a system. The decisions which product characteristics the basic system has to provide and which modules to be offered are supported by the results obtained using QFD-PPP.

Using the information gained in the workshop and through the satisfaction survey as basis, the plans for the next releases of the existing products were re-evaluated next. 71 potential product characteristics grouped into 12 groups were presented to the project team in another workshop and evaluated with regard to their fulfillment of the customer requirements.

	requirements (categorie)	functional requirement	level of satisfaction		satisfaction customer A	satisfaction customer B	comparison - competitor A		comparison - competitor B	
					(10=very good, 1= very poor)					
A	administration		6,43				6,43		5,86	
1	A.1	easy to use authorization scheme		6,5	6	7	8	1,5	5	-1,5
2	A.2	easy identification of users		7,5	6	9	9	1,5	5	-2,5
3	A.3	effort-saving software-deployment		5	5	5	5	0	7	2
4	A.4	easy (un)-installation of software-agents		6,5	5	8	4	-2,5	7	0,5
5	A.5	configurable system-parameters		7,5	6	9	7	-0,5	7	-0,5
6	A.6	high usability software-administration		6,5	5	8	8	1,5	5	-1,5
7	A.7	effort-saving release-updates		5,5	5	6	4	-1,5	5	-0,5

Fig.5. Customer Requirements including satisfaction and competitive analysis (excerpt).

As mentioned before, the requirements of the new customer segment were too inhomogeneous to be satisfied with a single additional software solution. At the same time the development team was unable to divide the segment into smaller customer groups which requirements could be met with a variant of the product portfolio. This was caused by the small pool of customers who could be asked about their requirements. So it seemed reasonable to start with a different approach: The software company decided to avoid developing a fixed number of variants of their software. Instead of this, a full version like before and a basic version of the product for the new customer segment should be offered. For this basic version a flexible upgrade path which allows customers to add several optional modules when needed was developed. Because customers in the software sectors are used to that kind of cafeteria system when licensing software, this approach seems promising. Figure 6 shows an excerpt of the identified modules for the basic and full version. So customers can upgrade their basic version step-by-step to a full version according to their actual needs.

Fig. 6. Comparison of the product versions (details blurred out for confidentiality).

5 Conclusion

It has been demonstrated how QFD-PPP can be used to identify different customer segments and their needs, to derive a product portfolio (i.e. members of a product line) systematically and derive common and variable product functions including exciting requirements that the customers would not have come up with. By identifying customer and the importance that these segments ascribe to different requirements, Value-Based Product Development is made possible. The results of the case studies and feedback gathered from participants show that QFD-PPP is well-suited to aid software companies in planning software product lines.

While we were not able to actually use cluster analysis as planned due to problems acquiring the number of potential customers necessary for the statistical analysis, we were able to demonstrate in the case study that a segment formed using traditional marketing methods was too broad and that a value-based analysis helped defining optional modules that can be offered to the customers. Collecting sales figures for these modules and having all new customers prioritize the requirements collected so far, clusters of customers can be formed over the next few years using real customers. Already with the small sample of three potential customers, we were able to detect heterogeneous requirements of customer groups before actually developing the software. This way, companies using QFD-PPP can avoid spending too much effort in developing software not suitable for the requirements of a customer segment.

References

1. Schmid, K.: Planning Software Reuse – A Disciplined Scoping Approach for Software Product Lines, Stuttgart: Fraunhofer IRB (2003)
2. Helferich, A., Herzwurm, G., Milcz, M. and Schockert, S.: Product Portfolio Planning using QFD, Proc. of the 14th International Symposium on QFD, Beijing (2008), pp. 334-343

3. Cohen, L.: Quality Function Deployment, Addison-Wesley, Reading, MA (1995)
4. Akao, J.: Quality Function Deployment: Integrating Customer Requirements into Product Design, translated by Glenn H. Mazur, Cambridge, MA (1990)
5. Chan, L.-W. & Wu, M.-L.: Quality function deployment - A literature review. European Journal of Operational Research, 143 (2002), pp. 463–497.
6. Herzwurm. G.; Schockert, S., Pietsch, W.: QFD for Customer-Focused Requirements Engineering, in: Proceedings of the 11th IEEE International Requirements Engineering Conference, Monterey Bay, USA (2003) 330-338.
7. Zultner, R. E.: Software quality function deployment – the North American experience, in: Software Quality Concern for people. Proceedings of the Fourth European Conference on Software Quality, Zürich (1994), pp. 143-158.
8. Herzwurm, G.; Schockert, S., Mellis, W.: Joint requirements engineering: QFD for rapid customer-focused software and Internet-development, Vieweg, Wiesbaden (2000)
9. Hauser, J.R., Clausing, D.: The House of Quality. Harvard Business Review, 66 (3), (1988), pp. 63-73.
10. Herzwurm, G., Schockert, S.: What are the Best Practices of QFD, in: Proceedings of the 12th International Symposium on QFD, Tokyo (2006)
11. Zultner, R.E.: Quality Function Deployment for Software: Satisfying Customers, in: American Programmer (1992), pp. 28-41.
12. Ohmori, A.: Software quality deployment approach: framework design, methodology, and example, in: Software Quality Journal No. 3 (1993), pp. 209-240.
13. Green, P.E., Krieger, A.M., Wind, Y.: Thirty Years of Conjoint Analysis: Reflections and Prospects, in: Interfaces, 31 (3), (2001), pp. 57-S73.
14. Gustafsson, A.: Customer focused product development by conjoint analysis and QFD. Dissertation Univ. Linköping, Dep. of Mechanical Eng., Div. of Quality Techn. (1996)
15. Fehlmann, T.: New Lanchester Theory for Requirements Prioritization, in: Proc. of the 2nd International Workshop on Software Product Management (2008), pp. 35-40.
16. Saaty, T. L.: Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World, 3rd edition, Pittsburgh, USA (1995)
17. Helferich, A., Schmid, K., Herzwurm, G.: Reconciling Marketed and Engineered Software Product Lines. In: Proceedings of the 10th International Software Product Line Conference, 23-27, Baltimore (2006)
18. Clements, P., Northrop, L.: Software product lines: practices and patterns. Addison-Wesley, Boston, MA (2002)
19. Bayer, J. et al.: PuLSE: A Methodology to Develop Software Product Lines, Proceedings of the 5th Symposium on Software Reusability (1999), pp. 122-131.
20. Weiss, D.M., and Lai, C.T.R.: Software product-line engineering: a family-based software development process, Boston (1999)
21. Herzwurm, G., Schockert, S., and Helferich, A.: QFD-PPP: Product Line Portfolio Planning using Quality Function Deployment, Proceedings of the 9th International Software Product Line Conference, Rennes (2005), pp. 162-173.
22. Ulwick, A.W.: Turn Customer Input into Innovation, Harvard Business Review, 80(1), (2002), pp. 91-97.
23. Stoermer, C., Bachmann, F., Verhoef, C.: SACAM: The Software Architecture Comparison Analysis Method. Technical Report CMU/SEI-2003-TR-006, Software Engineering Institute, Carnegie Mellon University (2003)
24. Herzwurm, G., Schockert, S., Breidung, M., Dowie, U.: Requirements Engineering for Mobile-Commerce Applications, in: Proceedings M-Business, Athens (2002)
25. Homburg, C., Krohmer, H.: Grundlagen des Marketingmanagements. 2nd Ed., Gabler, Wiesbaden (2009)

Knowledge-based Concretization of Requirements in Preparation for AHP

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. While planning a complex product, a huge number of requirements and weightings from several stakeholders are emitted. These requirements must be implemented into solutions in consideration of their weightings. The method Analytic Hierarchy Process (AHP) can be applied in order to prioritize the requirements structured. Therefore a hierarchy of requirements must be established. However, Stakeholders are only to a limited extent able to refine their requirements systematically during the planning process. Due to that the definition of a requirements hierarchy is difficult. A knowledge-based and computer-aided method for the concretization of requirements will be presented. Therewith stakeholders can refine their emitted requirements systematically which leads to the establishment of a requirement hierarchy. This serves as a preparation for the requirements prioritization in the framework of AHP.
Keywords: requirements concretization, AHP, hierarchy, ontology

1 Introduction

While planning a complex product (e.g. software) a high amount and variety of requirements and their weightings coming from different stakeholders are emitted. Thus, for example, the development of a computer-aided quality (CAQ)-software is a complex matter, because different industries and companies have specific requirements on a CAQ-software and there exists a multitude of application areas. Due to that, requirements and their weightings on software must be investigated carefully, in order to transform them into the optimal solution and thus to offer the optimal and stakeholder-oriented software-solution consisting of different application modules. Stakeholders of software are for example the buyer, the software engineer, system maintenance engineer and the user. Due to the complexity of planning process, a structured and traceable weighting of requirements is difficult to realize. There is a high risk that the emitted weighting of the whole amount of requirements are not consistent. The Analytic Hierarchy Process (AHP) is a method which expedites an effective decision-making process and which enables to check and to improve the consistence of weightings [1]. This method comprises several steps (fig. 1).

Fig. 1: Steps of AHP.

In the first step a goal has to be formulated that forms the top of the hierarchy [2]. In the case of software development the goal would be the establishment of a software according to the requirements of the stakeholders. After that the lower levels of the hierarchy must be generated which have to consist of sub-criteria and alternatives [3]. In the framework of product development sub-criteria have to be formulated in form of requirements and the alternatives in form of possible solutions. By means of paired comparisons weightings of the sub-criteria can be obtained [1]. Afterwards the consistence of the comparison can be checked [4]. The prioritization of alternatives forms the final step.

Accordingly, the requirements hierarchy forms the basis for the prioritization by using the method AHP [5][6][7][8]. Thereby, the concretization of the goal by dint of more concrete criteria is the most creative and crucial step during the appliance of AHP [3]. One problem in this case is that the gathered requirements are different regarding their level of concretization [9]. An establishment of a hierarchy of these requirements is therefore a complex matter. Outstanding expertise is necessary in order to build up a requirements hierarchy. After a stakeholder has formulated his requirements, it is not traceable for a third person (etc. the developer of the software) which of these requirements defines another requirement more precisely according to the stakeholder. The structuring of the whole amount of requirements into a hierarchy is therefore a complex and uncertain process. In order to solve the problem each stakeholder should define by himself which requirements do define his formulated requirement in more detail. This would create a requirement hierarchy for each stakeholder.

2 Existing Approaches for the Establishment of a Requirements Hierarchy

Approaches for the establishment of a requirements hierarchy already exist. One approach is the structuring of requirements into primary, secondary and tertiary requirements [10], whereby the degree of detail rises from a primary requirement to its tertiary requirements. One further approach for the concretization makes a distinction between the concretization of reference object, attribute and value of a requirement [11].

With these approaches a requirements hierarchy can be manually defined. However, an ordinary stakeholder is unpracticed in formulating and refining of requirements. He has no appropriate methods, in order to set requirements systematically [6]. Due to that a new computer-aided method for concretizing their requirements has been developed. This method identifies potential refining requirements of an emitted requirement and provides these to the stakeholder.

Thus the stakeholder can choose from these provided refining requirements those which are in sum sufficient to fulfill his emitted super-ordinated requirement. Therefore a high amount of knowledge about potential requirements as well as their potential detailing requirements must be available. On order to solve this problem a knowledge base has been created which can be accessed by the developed method during the automatic concretization of requirements. The knowledge base has been established in the form of an ontology which represents a formal description of knowledge by a set of concepts within a domain [12]. Within the ontology necessary knowledge can be modeled and becomes therefore machine-interpretable. In each individual application of AHP, a new requirements hierarchy has to be created [13]. Due to that the knowledge base can be utilized steadily with each planning process. The ontology is extendable and therefore its knowledge base increases with each additional planning process.

3 Creation of a Knowledge Base

An ontology consists of a class hierarchy in which the relevant terms are housed and consists of properties which are placed between the classes. A property specifies the unidirectional dependency between two ontology classes. The classes themselves contain a set of individuals, which can be named objects as well as instances [14]. Under each main-class a set of sub-classes have to be defined. Each of these sub-classes contains a term of a considered knowledge-domain.

In order to create a knowledge base for the concretization of requirements at first information components for the depiction of possible requirements were established. A typical requirement has a reference object, a property and a value [15]. The following figure 2 shows an example of a requirement which has been divided up into its three components.

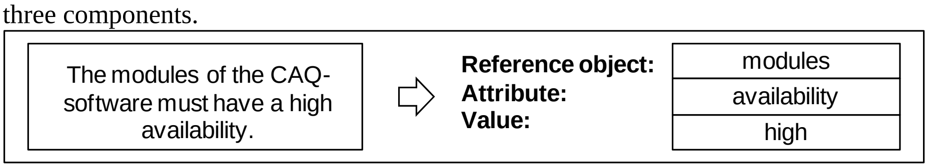


Fig. 2: Requirement components.

In this way an amount of potential requirements were defined and modeled within the ontology by using the classes “reference object”, “attribute” and “value” as well as the properties “has attribute” and “has value” (fig. 3). “Has_attribute“-properties connect the sub-classes of the „reference object“-hierarchy with the sub-classes of the „attribute“-hierarchy. They are needed to specify which attributes each reference object can potentially have. The “has_value”-properties are defined between “attribute”-sub-classes and “value”-sub-classes. This kind of properties specifies which values an attribute can assume. The mentioned properties define only sensible connections between the classes and subclasses. These combinations comprise all potential requirements that can occur during the collection of requirements.

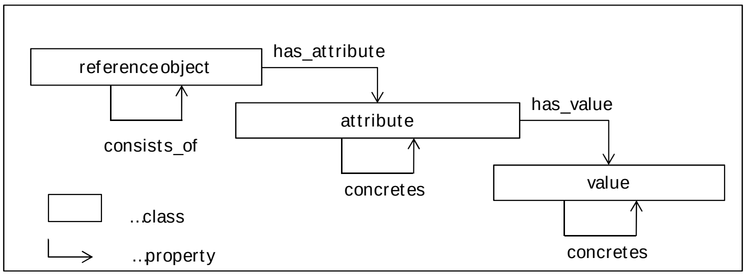


Fig. 3: Schematic overview of the ontology.

Also knowledge about which requirement component concretes potentially another component is modeled within the ontology. This knowledge is on the one hand stored by means of the structure of class hierarchy. In this regard, each term of a class must have a „is a“-relation to the term of its super-ordinated class-, which means that each sub-term must be a sub-form of the super-ordinated class term. And on the other hand, the knowledge is defined by different properties. Concerning the “reference objects”, these are the “consists of” properties which are set between two sub-classes of the class hierarchy “reference object”. They define which reference object could potentially consists of another reference object. In order to store knowledge about the concretization of attributes „concretes“-properties were defined between the classes of the „attribute“-hierarchy. With reference to the concretization of values, “concretes”-properties were set between classes of the “value”-hierarchy. In the following, cut-outs of an ontology belonging to the domain CAQ-software are presented.

Class Hierarchies

The reference object-hierarchy for the product “CAQ-Software” (fig. 4) comprises the “CAQ-Software” as a whole, its modules such as “First sample examination module” and its components like the “Presentation layer”. The second upper class is named „Attribute“. Its subclasses are for example the “Availability” and the “Inspectability”. The third upper class is the demanded “Value” of the attribute. The required value of the reference object’s attribute can be differentiated into qualitative (e.g. “High”, “Low”) and quantitative value.

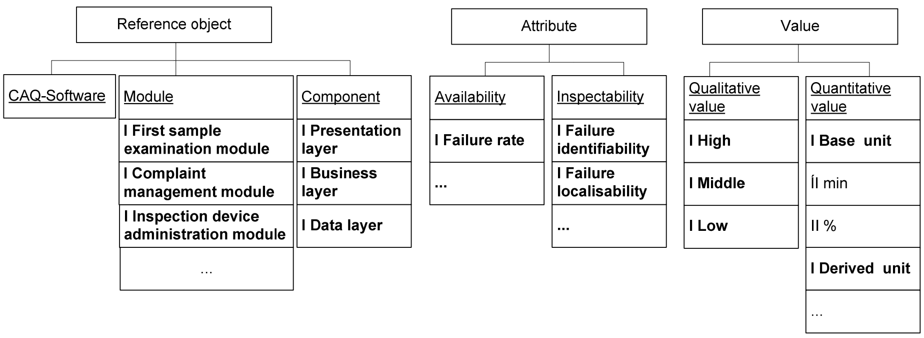


Fig. 4: Class-hierarchy.

Properties

In figure 5 an ontology-cut-out is shown.

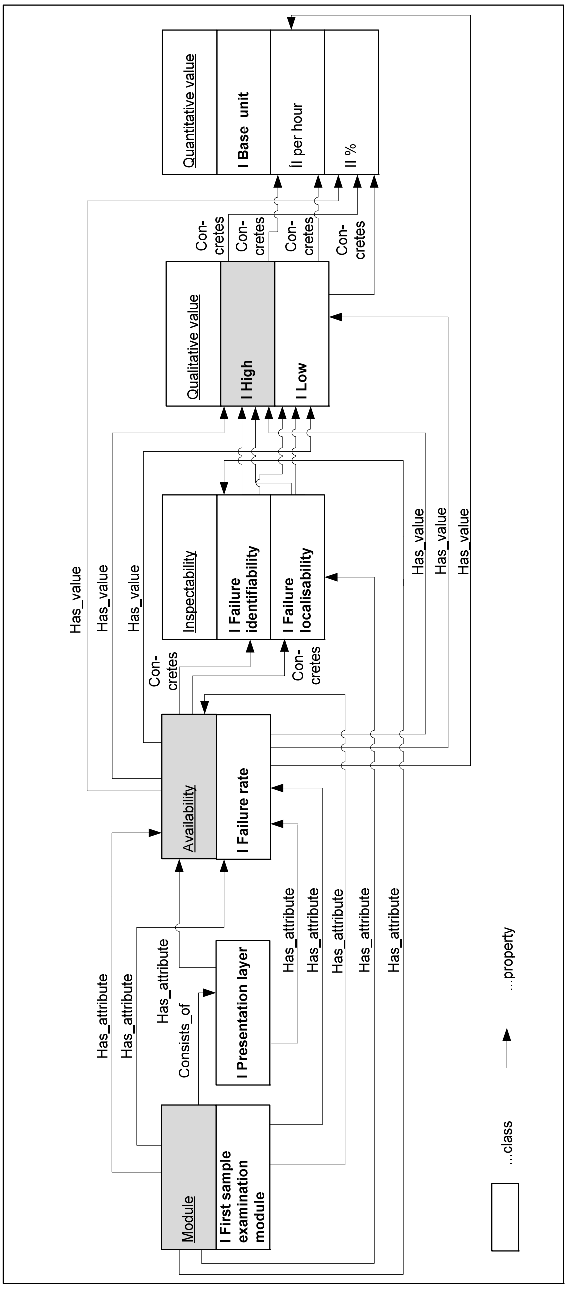


Fig. 5: Example for the defined properties.

This out-cut comprises properties which are defined between the three relevant classes (“Module”, “Availability” and “High”) of our given example in figure 1 and other classes of the ontology. Over the “Has_attribute”-properties it is defined which attributes a module of a software can have. By means of the “Has_value”-properties it is defined which values the attribute “Availability” can assume. The relations between the reference objects were defined by using a “consists of” property. A “module” consists e.g. of a “Presentation layer”. Also “concretes”-properties are defined between the components attribute and value of our given example and the other ontology-classes.

4 Knowledge-based Method for Concretization

During the planning process stakeholder utter their requirements in natural language. In order to put the requirements in relation to the knowledge of the ontology the requirements have to become highly formalized. During their formalization, a requirement is divided into its information components (fig. 1). After the formalization the information components are mapped structured in a “requirement template” [15].

During the appliance of the developed method the stakeholder can choose between the concretization of the reference object, the attribute and the value of his considered requirement. In case of the concretization of reference object the user can also chose if there should be a “is a” or a “consists of” relation between the reference object of the given requirement and the reference objects of the provided detailing requirements. The following procedure describes how the method identifies automatically detailing requirements of a given requirement on the basis of the ontology (fig. 6).

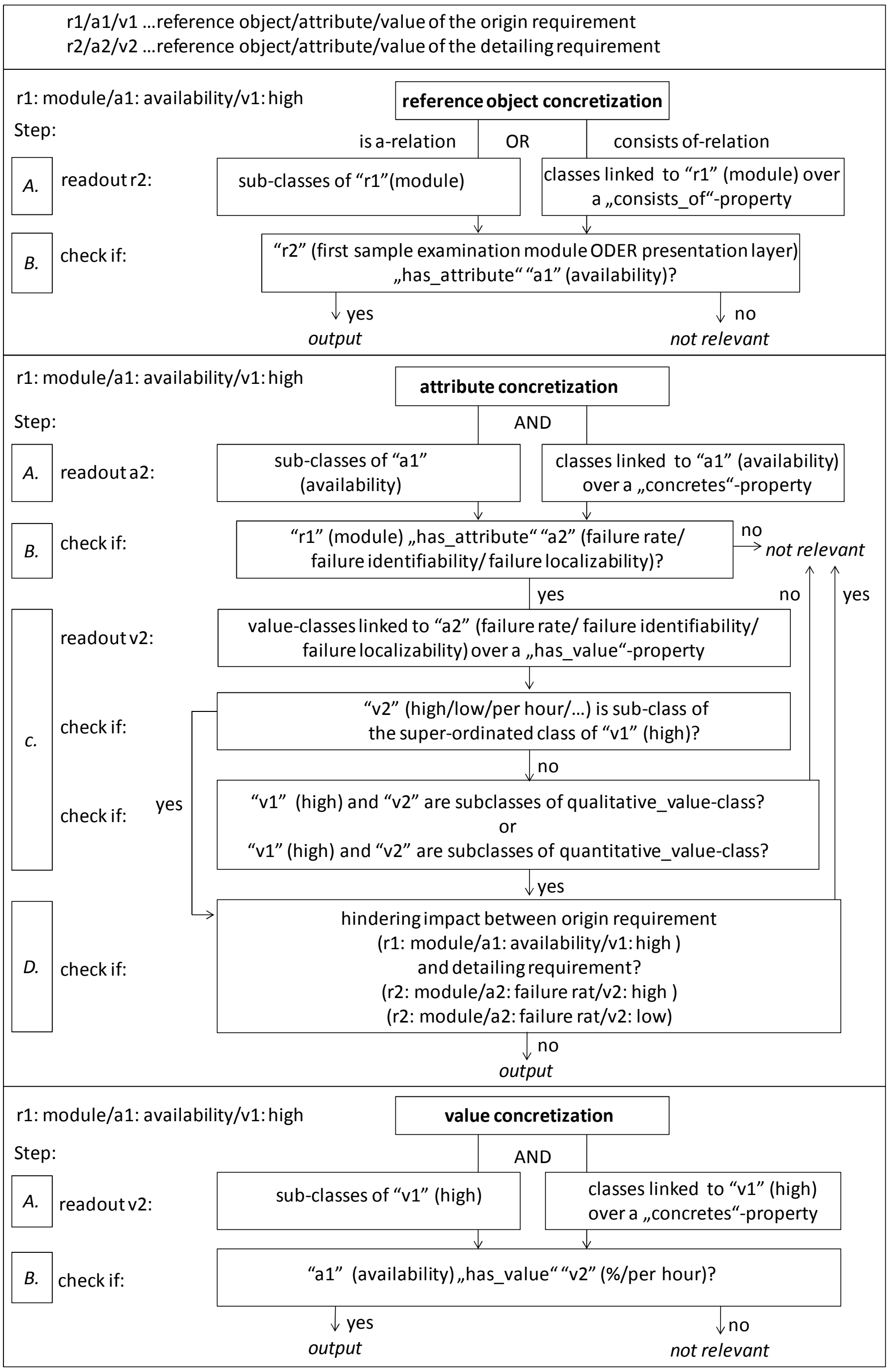


Fig. 6: Procedures of the method for concretization.

Reference object concretization

During the reference object concretization the reference object of an origin requirement becomes more concrete and the attribute as well as the value remain the same. At first (step A) the method checks if the stakeholder demands a „is a"- or a „consists of"-relation between the origin reference object and the detailing reference objects. A „is a"-relation between two reference objects is defined by the structure of the reference object-class-hierarchy. If the stakeholder chooses this kind of relation, all reference objects which are a sub-class of the reference object-class of the origin requirement will be identified. In the case of the given example (fig. 1) this would be the reference object "first sample examination module" (fig.5). If the stakeholder decides for a „consists of"-relation, the method detects all reference object-classes to which a „consists_of"-property, coming from the origin "reference object"-class, points. Considering the given example, a "consists_of"-property points to the class "presentation layer" (fig. 5).

In order to provide only sensible requirements to the stakeholder, the method checks for each refining reference object from step A, whether it could potentially assume the attribute of the origin requirement (**step B**). This is the case with reference objects, whose ontology-class is connected with the attribute-class of the origin requirement over a „has_attribute"-property. The detailing reference objects of the given example do comply with this condition and can be therefore provided to the stakeholder (fig. 7). These detailing requirements are potentially suitable to build up a new level within the AHP requirements hierarchy.

Origin requirement:	Modules	Availability	**I High**
Detailing requirement („Is a"-relation):	**I First sample examination module**	Availability	**I High**
Detailing requirement („Consists of"-relation):	**I Presentation layer**	Availability	**I High**

Fig. 7: Results of the reference object concretization.

Attribute concretization

During the attribute concretization the attribute is described more detailed (Humpert95) and the reference object remains the same. Although the value stays on the same abstraction level as the origin value, in some cases the value has to change, in order to support the goal of the super-ordinated requirement. In the first step (step A), the method identifies all attribute-classes within the ontology which are a sub-class of the super-ordinated attribute-class of the considered requirement or which are connected with the super-ordinated attribute-class over a "concretes"-property. For our given example these are the attribute classes "failure rate" (sub-class) and "failure identifiability" as well as the "failure localizability" (concerning the connection over "concretes"-properties).

In order to obtain a meaningful output the method checks (**step B**) whether the reference object of the origin requirement can have the more concrete attributes which were identified within step A. Therefore, it will be checked whether these attribute-classes are connected over a “has_attribute”-property to the “reference object”-class of the abstract requirement. This precondition is fulfilled concerning the attributes identified in step A of the given example (fig. 5). In order to ensure the achievement of the objective of the super-ordinated requirement, the value of the detailed requirement must perhaps change. In this case, the value of the detailing requirement remains on the same abstraction level of the origin value.

In order to identify an appropriate value the method (**step C**) determines the value-classes to which each detailing attribute is linked over a „has_value“-property. For example, the attribute “failure rate” is linked with the value-classes “high”, “low” and “per hour”. Afterwards it has to be checked if these identified value-classes are sub-classes of the super-ordinated class of the origin value-class (“high”). Values which comply with this are on the same abstraction level as the origin value. For example the classes “high” and “low” are sub-classes of the super-ordinated value class of the origin value.

If this is the case, it has to be checked (**step D**) if there exists a hindering impact between the origin requirement and the detailing requirement. This can be recognized by applying a method for the identification of hindering impacts between requirements. For an in-depth description of this method see [16]. If e.g. the requirement on a high failure rate is implemented this has a negative impact on the realization of the requirement on a high availability. In this case a high failure rate is inappropriate for detailing the origin requirement. If there is no hindering impact between two considered requirements, then the detailing requirement can be presented to the stakeholder. There is no negative impact between the origin requirement and the requirement on a low failure rate. Thus, the requirement on a low failure rate can be provided to the stakeholder (fig. 8). Anyway, there occur also detailing attributes whose potential values are not a sub-class of the super-ordinated class of the origin value-class. In this case, for each value has to be checked whether the detailing value and the value of the origin requirement are part of the “qualitative value”-class-hierarchy or if both classes are part of the “quantitative value”-hierarchy. Therewith it can be checked if both values are located on the same level of abstraction. Values which are on the same level as the origin value must be analyzed concerning their potential hindering effects on the super-ordinated requirement. Only if there is no hindering impact, the detailing requirements can be made available to a stakeholder. For our given example the method identifies a set of the more concrete requirements shown in figure 8. All of them could be potentially part of the next lower level of the AHP requirements hierarchy.

Origin requirement:	Modules	Availability	I High
Detailing requirement:	Modules	**I Failure rate**	**I Low**
Detailing requirement:	Modules	**I Failure identifiability**	**I High**
Detailing requirement:	Modules	**I Failure localisability**	**I High**

Fig. 8: Results of the attribute concretization.

Value concretization

Only the value of a given requirement changes during the automatic value concretization. This kind of concretization does not lead to the establishment of a new level of AHP-requirements hierarchy. Rather, the provided detailing value substitutes the origin value of a requirement within a given hierarchy level. This enables the specification of requirements values within a given hierarchy level. In step A, the method indicates all „value"-classes that are a sub-class of the origin value-class or that are connected to the origin value-class over a "concretes"-property. In the given example coming from the value-class "high" a "concretes"-properties point to the value-classes "%" and "per hour". Afterwards (step B), it is checked whether the attribute of the origin requirement could potentially assume the refining values, which were determined in step A. This is fact when a "has_value"-property coming from attribute-class of the origin requirement points to the refining value-classes. Only these values, which comply with the mentioned condition, will be presented to the stakeholder. In the event that it is a quantitative value, only the unit of a value will be provided to the stakeholder. The stakeholder must add an appropriate numerical value to the unit, in order to complete the value of the refining requirement. Considering the given example, a "has_value"-property is defined between the classes "high" and "%". Consequently, the detailing requirement "Modules", "Availability", "per hour" is no appropriate requirement in order to detail the sub-ordinated requirement. Just the detailing requirement shown in figure 9 can be provided to the stakeholder.

Origin requirement:	Modules	Availability	**I High**
Detailing requirement:	Modules	Availability	II %

Fig. 9: Results of the value concretization.

Selection and prioritization of appropriate refining requirements

While the refining requirements were provided to a stakeholder he has to select those requirements which are in sum adequate, to fulfill their super-ordinated requirement. Afterwards, the refining requirements will be stored. The concretization of reference object and the concretization of attribute leads to the establishment of a new requirements hierarchy level whereas the value concretization formulates the values of requirements within a hierarchy level more precisely. During the next step of application of AHP alternatives in form of possible solutions must be added to the requirements hierarchy.

Afterwards, requirements which are on the same level will be prioritized by means of paired comparisons [1]. A stakeholder could for example choose all of the provided detailing requirements shown in figure 5 in order to refine his origin requirement “The modules of the CAQ-software must have a high availability.”. After this, he has to prioritize this three detailing requirements in the framework of paired comparison shown in figure 10. Within the given paired comparison matrix the value 9 expresses an extreme preference and the value 1 defines an equal preference of a requirement compared to another requirement. According to the considered stakeholder, the requirement for a low failure rate is e.g. much more important than the realization of the requirement for high failure localizability. For this reason the defined value within the matrix is 7.

			Modules	Modules	Modules
			Failure rate	Failure identifiability	Failure localisability
			Low	High	High
Modules	Failure rate	Low	**1**	5	7
Modules	Failure identifiability	High	**1/5**	**1**	**3**
Modules	Failure localisability	High	**1/7**	**1/3**	**1**

Fig. 10: Paired comparison matrix.

This mentioned procedure has to be applied to each level of the AHP requirements hierarchy in order to make a ranking of the whole amount of requirements. Finally, each stakeholder has created his own requirements hierarchy that comprises prioritized requirements. These hierarchies coming from the stakeholders have to be merged into one hierarchy by means of statistical approaches [1]. Subsequent, a prioritization of alternatives can be obtained and thus solutions which can best implement the given requirements have been identified.

5 Summary and forecast

The work presented herein describes a computer-aided method for the knowledge-based construction of a requirements hierarchy. This method automatically provides requirements which can formulate a considered requirement more precisely to the stakeholder. Due to that the stakeholder has the possibility to substitute his abstract requirement by all or by several of these detailing requirements. This allows an optimization of expressing requirements and enables a secured deriving of concrete requirements. It becomes traceably which concrete requirements must be actually fulfilled in order to fulfill the super-ordinated requirement. An ontology serves as the knowledge base for the presented method. This ontology contains knowledge about potential requirements and knowledge of which requirements can potentially refine another requirement. Due to the method the whole amount of requirements can be captured and structured in form of a hierarchy. All in all the method supports the appliance of AHP because the quality of the hierarchy can be improved.

During the construction of a hierarchy for the appliance of AHP it has to be secured, that the criteria being on the same level should not have relationships between each other [17]. But in most cases an independency of criteria cannot completely ensured. Extended approaches for the handling of dependencies during the appliance of AHP already exist. However, due to the high amount of requirements there is a high risk that dependencies between requirements remain undiscovered. The above mentioned method for the detection of impacts between requirements allows the detection of hindering and supporting dependencies between requirements. It has to be investigated in the future if this method is sufficient or adaptable in order to identify all relevant dependencies between the requirements during the appliance of AHP. Moreover, it must be clarified how the different forms of requirements dependencies should be handled during the appliance of AHP.

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. Saaty T.-L., Vargas L.-G.: Models, methods, concepts & applications of the analytic hierarchy process, Springer, 2000.
2. Xie, M., Tan, K.C., Goh, T.N.: Advanced QFD Applications, United States: ASQ Quality Press, 2003, pp. 216.
3. Navneet B.: Strategic decision making : Applying the analytic hierarchy process / Navneet Bhushan and Kanwal Rai. - London : Springer, 2004. - IX, 172 S. : graph. Darst. : 24cm. - (Decision engineering) ISBN 1-85233-756-7
4. Daniel J. Power: Decision Support Systems: Frequently Asked Questions
5. Saaty, T.: How to make a decision: The Analytic Hierarchy Process. European Journal of Operational Research Nr. 48: S. 1990.
6. Hepler, C.; Mazur, G.: „Finding customer delights using QFD";Tagungsbeitrag: 18th Symposium on QFD; 2006.
7. Refflinghaus R.: Einsatz des Analytischen Hierarchie Prozesses zur Vorbereitung der kundenspezifischen Eingangsgrößen eines Quality Function Deployments, Sonderforschungsbereich 696: Forderungsgerechte Auslegung von intralogistischen Systemen– Logistics on DemandTechnical Report 0901, ISSN 1867-3473, 2009.
8. Childs D., Serino D., Bartlett M., Stover S.: DREAM/QFD to Re-design Staff Service Excellence at Rutland Regional Hospital Systems, In Proceedings of the 16th International Symposium on QFD, Portland/USA, 24.09.-25.09.2010.
9. 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.
10. Kamrani, A.-K., Nasr E.A.: Engineering Design and Rapid Prototyping, New York: Springer, 2009.
11. Jung, C.: Anforderungsklärung in interdisziplinärer Entwicklungsumgebung, München: Dr. Hut, 2006.
12. Maletz M.: Integrated requirements modeling: A contribution towards the integration of requirements into a holistic product lifecycle management strategy, Kaiserslautern: Technische Universität Kaiserslautern, ISBN 978-3-939432-92-0, 2008.
13. Naumann F.: Quality-driven query answering for integrated information systems, Number 2261 in Lecture Notes in Computer Science. Springer. 2002.
14. Horridge M., Knublauch., Rector A., Stevens R., Wroe C.: Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools. Edition 1.0. The University Of Manchester. 27. August 2007.
15. Klute S., Kolbe C., Refflinghaus R.: Data Processing System for structured Capturing and Analyzing of Requirements. In: Simone Bürsner, Jörg Dörr, Andreas Gehlert, Andrea Herrmann, Georg Herzwurm, Dirk Janzen, Thorsten Merten, Wolfram Pietsch, Klaus Schmid, Kurt Schneider, Anil Kumar Thurimella (Eds.): 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, Essen, October 2010, pp. 87-99.
16. Crostack, H.-A.; Kolbe, C.; Refflinghaus, R.: Computer-aided method for automatic identification of effect relations between requirements on an intra-logistics facility. In Proceedings of the 16th International Symposium on QFD, Portland/USA, 24.09.-25.09.2010
17. Meixner O., Haas R.: Wissensmanagement und Entscheidungstheorie, Facultas, Wien, 2010.

Construction and Evaluation of an Algorithmic and Distributed Prioritization Method

Annabella Loconsole[1,3], Hannes Gruber[2], Adrian Nae[2], Björn Regnell[3]

[1] Department of Computer Science, Malmö University, Sweden
[2] Flygprestanda AB, Malmö Sweden
[3] Department of Computer Science, Lund University, Sweden
Annabella.Loconsole@mah.se
{ic05hg7, c03an}@student.lth.se, {Björn.Regnell, Annabella.Loconsole}@cs.lth.se

Abstract. Prioritization of requirements and tasks is an important activity of the development process; it can however be a very complex and time-consuming activity. In this paper we present MAAD (Method for Agile, Automated and Distributed prioritization) an algorithmic prioritization method. The method was developed at Flygprestanda AB, which needed a systematic and automated way of prioritizing requirements and work tasks. The method is constructed based on existing prioritization methods, focusing on minimizing the end-user time-consumption. Factors considered in the priority calculation are Urgency, Benefit, Cost, and Type. Furthermore, in order for the method to suit different companies and projects the importance of the different factors is adjustable. An evaluation was performed, which showed that MAAD prioritization was easier-to-use, less time-consuming, more accurate, more scalable and the prioritization method most suitable for Flygprestanda compared to Wiegers method and the prioritization method used at the company.
Keywords: prioritization, factor, weight, algorithm**,** urgency, benefit, cost and type

1 Introduction

Which is the most important task at hand? This is the central question that a prioritization method aims at answering. Prioritization must be a part of every decision-making process. In the context of requirements engineering, prioritization usually means deciding the order in which requirements shall be implemented. Widening the perspective, the same principles hold for any form of work, every kind of work task must be evaluated and planned accordingly. Exactly how the prioritization is done varies between different persons and companies. Usually it is done through subjective judgments of the factors that affects, and are affected by, the decision at hand.

Prioritization can however be very time consuming, especially when dealing with large amounts of information. During the prioritization process both short-term and long-term effects of information items such as requirements, tasks and requests must be evaluated. This is particularly complicated if the items affect each other.

Another issue when performing the prioritization is that the knowledge required to prioritize usually is distributed among several persons, for instance employees and customers.

These challenges lead to the following research questions: 1. Is it possible to reduce the individual workload by introducing an automated and distributed prioritization method for requirements, tasks and change requests? 2. Can an automated and distributed prioritization method increase the quality of a company's software and services?

In order to answer these questions, we constructed a Method for Algorithmic, Automated and Distributed prioritization (MAAD) at Flygprestanda AB. The method is based on existing prioritization methods, with the focus on minimizing the end-user time-consumption. Factors considered in the priority calculation are Urgency, Benefit, Cost and Type and in order for the method to suit different companies and projects, the importance of the different factors is adjustable. An evaluation was performed with the purpose of comparing MAAD to Wiegers method [3] and the prioritization method used at Flygprestanda. The evaluation showed that MAAD prioritization, was easier-to-use, less time-consuming, more accurate, more scalable and the prioritization method most suitable for Flygprestanda.

The paper is structured as follows: in the next Section we briefly describe different requirements prioritization methods, in Section 3 it is described the context of this study. In Section 4 we present our prioritization method. An evaluation is described in Section 5. Finally discussion and conclusions are presented in Section 6.

2 Requirements Prioritization Methods

In this section we describe several techniques that are available in the literature used to prioritize requirements. An overview of prioritization factors can be found in [1].

The Analytic Hierarchy Process (AHP) was first introduced by Saaty [16]. The basic idea of this method is to determine a ratio scale[3] list of items by comparing all possible pairs (pair-wise comparison). The comparison complexity of $n(n-1)/2$ which results in 45 comparisons for 10 requirements, is the major disadvantage of the AHP method, but can be lowered by exploiting machine learning techniques [6], [7], [8].

Case-Based Ranking (CBRank) is a method similar to AHP that was designed specifically to use machine learning techniques to overcome the scalability problems in AHP. Another difference with CBRank is that the user has to specify a strict preference, i.e. decide which of the compared items is the most important [7], [8], [9]. Other methods that use pair-wise comparisons are Bubblesort that has the same comparison complexity as AHP and Binary Search Tree (BST) data structure [10].

In the Cumulative voting technique, each stakeholder gets an amount of imaginary money or points that can be spent on the different requirements, tasks or requests that are subject to prioritization. This technique is used in the Hundred-dollar-test where all stakeholders are given 100 imaginary dollars to distribute among requirements [11]. Variations have been tested with larger amounts of money to better suit prioritization of larger sets of requirements [12].

[3] See [5] for a definition of nominal, ordinal, interval, and ratio scales.

When performing prioritization using cumulative voting, an issue could be that stakeholders can use different voting strategies and tactics.

Table 1. Comparison of prioritization methods.

Method	Technique(s)	Factors considered	Result type	Reference
AHP	Pair-wise comparisons	Priority	Ratio	[16]
Binary Search Tree	Pair-wise comparisons	Priority	Ordinal	[14]
Bubblesort	Pair-wise comparisons	Priority[4]	Ordinal	[14]
CBRank	Pair-wise comparisons	Cost, benefit[5]	Ratio	[7]
Hundred-dollar-test	Cumulative voting	Priority	Ratio	[11]
MoSCoW	Numerical assignment	Priority	Nominal	[15]
Planning game	Numerical assignment, ranking	Effort, priority	Ordinal	[4, 13]
Top-ten	Ranking	Priority	Nominal	[2]
Wiegers method	Numerical assignment	Benefit, penalty, cost, risk	Ratio	[3]
MAAD	Numerical assignment	Urgency, Benefit, Cost, Type	Ratio	[1]

Numerical assignment is a common technique where items are put into different priority groups. Often three or four different groups are used, such as Low, Medium, High or as in the MoSCoW method: Must have, Should have, Could have, Won't have. Numerical assignment is used in eXtreme Programming as part of the Planning game when the contents of the next release are planned by the developers and customer [13]. The items, called stories, are placed in different piles according to priority and risk and can thereafter be more easily ranked within the groups [4]. The Top-ten method consists in ranking only the 10 most important items at the time are identified, the rest are left until the next time the prioritization is performed [2]. Wiegers method also uses numerical assignment for estimation of benefit, penalty, cost and risk [3]. The different factors are estimated for every item and are assigned a value ranging from 1 to 9. The priority score is calculated according to the following formula:

$$Priority = \frac{Benefit\% * Benefit\ weight\ +\ Penalty\%\ *\ Penalty\ weight}{Cost\%\ *\ Cost\ weight\ +\ Risk\%\ *\ Risk\ weight}$$

Table 1 summarizes the described methods. It also includes our MAAD prioritization, which is presented in section 4. Among these, only the Wiegers method can be applied in the company Flygprestanda since it considers several factors. Therefore it will be compared with MAAD during the evaluation.

[4] The method supports any factor but only one at a time.
[5] The method can combine up to two factors at a time.

3 Context

The method described in this paper has been developed for the company Flygprestanda AB, which is specialized in airport analyses and performance calculations within the aviation industry. The company has currently about 60 employees whereof half of these are software developers. The typical customer is a small to medium sized airline company, which relies on the services provided by Flygprestanda for their daily operations.

Currently about 7000 to 10000 requirements and change requests are handled each year, distributed over a few hundred software and service projects, but these numbers will increase as the company grows. The work process at Flygprestanda is agile in the sense of iterative work, relatively short release cycles and a clear focus on producing code rather than extensive documentation [1]. Differently from the waterfall development, iterative work demands regular re-prioritization.

The current prioritization method at Flygprestanda is ad hoc. The customer prioritize themselves their requests and error reporting. The problem is that almost every request and report is assigned the highest priority by the customer. This causes distrust in the set priorities and it makes it hard to recognize which request that should be handled first. This is a widespread and well known problem in requirements prioritization resulting in disbelief and in limited usefulness of the set priorities [2], [3].

Our prioritization method MAAD, described in the next section, will be integrated in Scope which is a new software system recently introduced in the company. Scope is developed as an in-house project whose main objective is to help manage and plan company projects. The system also handles the information flow within the company as well as between the company and its customers. The system is built to handle *nodes*, a data structure which serves as a container for different kind of information. The information could for instance specify requirements, tasks, and change requests. A *project* is defined as a set of nodes, which can change over time. Every project has its own set of nodes.

4 The Prioritization Method MAAD

According to the company's needs, the prioritization method shall be implemented as an algorithm in Scope and be able to handle several factors. It shall also allow for adjustments since the importance of different factors to different company departments or projects might vary.

The method that we have created has been integrated in Scope, which is a web based client-server application. The priority calculation is performed on the server-side by the algorithm before a node is sent to the client. The graph in Figure 1 visualizes the priority-related data flow in Scope. The data flow paths are explained in the following list.

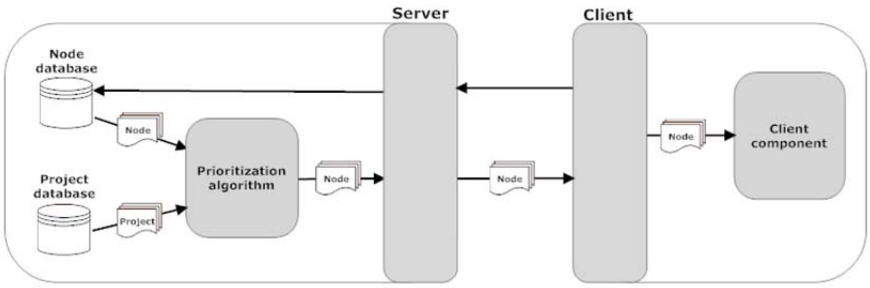


Fig. 1. Priority-related data flow in Scope when a client requests nodes from the server.

1. The client sends a request to get a set of nodes from the server.
2. The nodes are extracted from the node database and every node is, in turn, handed to the prioritization algorithm.
3. The algorithm retrieves parameters from the node (please refer to [1], chapter 3 for a definition of parameters).
4. The algorithm requests the data structure of the project which the node belongs to using the node parameter QM number (the quality management number that identifies the node project).
5. The node project is extracted from cache or from the project database.
6. The algorithm collects parameters from the project.
7. The algorithm calculates and sets the priorities parameters in the node data structures.
8. The set of nodes is thereafter sent to the client.

The stages 3-7 in the above list are repeated once for every extracted node. No significant negative performance impact due to integration of the prioritization algorithm has been experienced at the time of writing.

Table 2. Factors and weights.

Factors and weights	Based on parameter(s)
Urgency factor	Effort, Deadline date or Target date, Current date
Benefit factor	Severity or Value
Cost factor	Effort
Type factor	Type, Enhancement weight, Defect weight, NCR[6] weight, Task weight
Urgency weight	Urgency weight, CB weight
Benefit weight	Benefit weight, Cost weight
Cost weight	Benefit weight, Cost weight
CB weight	Urgency weight, CB weight

[6] NCR means Non-Conformity Report. A report used to describe a serious fault and the immediate and preventive actions taken to correct it.

Our algorithm calculates node priority based on four main factors; Urgency, Cost, Benefit and Type. The algorithm is made highly adjustable by the use of different weights. The factors and weights used by the prioritization algorithm are presented in Table 2 as well as the node and project parameters that they are based on. To distinguish factors, weights, parameters, constants and variables the following notation will be used:

- Factors are named Fx, where x is the name of the factor.
- Weights are named Wx, where x is the name of the factor that the weight corresponds to.
- Parameters are named Px, where x is the name of the parameter.
- Constants are named with unique lowercase letters.
- Variables are named with unique capital letters.

The following formulas are used to calculate the priority of a node. These formulas are based on the design decisions and on the literature study both described in [1], where the details of the algorithmic prioritization method can also be found; for limited space reasons a brief overview is given here:

Table 3. Formulas used to calculate the prioritization.

Formula Num	Formula
1	$Priority = \left(F_{Urgency} * W_{Urgency} + \frac{F_{Benefit} * W_{Benefit}}{F_{Cost} * W_{Cost}} * W_{CB}\right) * F_{Type}$.
2	$F_{Urgency} = \frac{c_1}{c_2 + b * H^2}$.
3	$F_{Benefit} = P_{Severity}\ or\ P_{Value}$.
4	$F_{Cost} = k * P_{Effort} + m$.
5	$W_{Urgency} = \frac{P_{Urgency\ weight}}{P_{Urgency\ weight} + P_{CB\ weight}}$.
6	$W_{CB} = \frac{P_{CB\ weight}}{P_{Urgency\ weight} + P_{CB\ weight}}$.
7	$W_{Benefit} = \frac{P_{Benefit\ weight}}{P_{Benefit\ weight} + P_{Cost\ weight}}$.
8	$W_{Cost} = \frac{P_{Cost\ weight}}{P_{Benefit\ weight} + P_{Cost\ weight}}$.
9	$F_{Type} = \frac{T}{P_{Enhancement\ weight} + P_{Defect\ weight} + P_{NCR\ weight} + P_{Task\ weight}} + c$.
10	$H = Work\ hours\ until\ Deadline\ or\ Target\ date - Effort$.
11	$c = 1 - \frac{1}{number\ of\ types} = 1 - \frac{1}{4} = 0.75$.

Urgency factor. The Urgency factor in formula 2 is based on the multi-user specified node parameter Effort, the user specified Deadline or Target date and the Current date. Every Effort choice has a corresponding numerical value, which is the matching number of work hours. The choices are presented below, with their numerical values in parenthesis. The default value, which is used in the calculation if no user has estimated Effort, is presented in bold.

1 day (8), 2 days (16), 3 days (24), 1 week (40), 2 weeks (80), 3 weeks (120) 1 month (160), 2 months (320), 3 months (480)

The variable H, defined as the remaining time (in work hours) to when work has to begin, is calculated according to the principle shown in formula 10 in Table 3.

The Deadline date is used to calculate H if it is before the Target date. The Target date is used to calculate H if it is before the Deadline date. When one or more users have estimated Effort, the average of all estimations is used in the calculation in formula 10. The calculation of course has to be adjusted to consider the number of work hours per workday, the number of workdays per work week and so on. A lower limit of 0 is enforced on the H value to ensure that the Urgency factor in formula 2 stays at its maximum value when Deadline or Target date is passed. The priority difference between items that have passed their deadline or target dates will therefore entirely depend on their benefit, cost and type factor values.

The ratio c_1/c_2 specifies the maximum resulting value of formula 2 and this ratio, in combination with b, decides how fast the value will increase when H approaches 0. The actual values of c_1, c_2 and b used in the prioritization algorithm are presented below.

$$c_1 = 10000,\ c_2 = 1000,\ b = 0.5\ .$$

These values were chosen to limit the maximum value to 10 and to achieve a priority increase curve. This results in a period of approximately two weeks, where priority increases. Two weeks, which is a short term planning, correspond to the iteration length of the iterative development of the company. This should be well enough for an employee to change focus and switch work task.

Benefit factor. The Benefit factor in formula 3 in Table 3 is based on the user specified node parameter Severity or Value depending on the node parameter Type. The Severity and Value choices have corresponding numerical values, which are used in the priority calculation in formula 1. The choices are presented below, with their numerical values in parenthesis and the default choice presented in bold. Severity choices for flight-specific projects and node types Defect and NCR are the following:

• **None (1),** • No Op impact (2), • Minor Op impact (3),
• Major Op impact (5), • Ac on ground (7), • Fleet on ground (10)

Severity choices for non-flight-specific projects and node types Defect and NCR are the following:

• **Minor frustration (1),** •Major frustration (3), •Minor functionality loss (5),
• Major functionality loss (7), • Complete functionality loss (10)

Value choices for all projects and node types Enhancement and Task are the following:

• **Trivial improvement (1)** • Minor improvement (3), • Normal improvement (5), • Major improvement (7), • Essential improvement (10)

The values we have chosen for the different Severity and Value levels represent the exponential increase of criticality or value that is the “general opinion” of the Severity and Value levels at Flygprestanda.

Cost factor. The Cost factor in formula 4 in Table 3 is based on the multi-user specified node parameter Effort. The Effort choices and their corresponding numerical values are described above (see the description of Urgency factor) and once again it is the average of the estimations that is used in the calculation unless no estimations have been made. In that case the default value is used. The Cost factor is calculated using a linear equation where k decides how fast the score increases and m decides the starting value. The actual values of k and m used in the prioritization algorithm are presented below.

- $k = 0.003$
- $m = 1$

These values were chosen to limit the maximum value of the Cost factor to approximately 2.5. This is the case when the Effort average is at its maximum (480 hours). This limit was introduced in Scope to prevent high effort times to decrease the combined Cost-benefit priority to such a low level that items are neglected.

Urgency and CB weights. The Urgency weight in formula 5 and the CB weight (Cost Benefit weight) in formula 6 are based on the user specified project parameters Urgency weight and CB weight. The weights are used in the priority calculation in formula 1 to achieve a suitable balance between main factor Urgency and the combined main factors Benefit and Cost.

The parameter choices have corresponding numerical values that are used when calculating the weights. The choices are presented below, with their numerical values in parenthesis and the default choice (for both Urgency weight and CB weight) presented in bold.

• Very low (0.2), • Low (0.35), **• Medium (0.5)**, • High (0.65), • Very high (0.8)

The maximum values of formula 5 and formula 6 are thus 0.8 and the minimum 0.2. This means that it is not possible to totally disregard a main factor. The range of 0.2-0.8 is specific for the implementation in Scope.

Benefit and cost weights. The Benefit weight in formula 7 and the Cost weight in formula 8 are based on the user specified project parameters Benefit weight and Cost weight. The weights are used in the priority calculation in formula 1 to achieve a suitable balance between the main factors Cost and Benefit. The default and available parameter choices and their numerical values are the same as Urgency and CB weights for both Benefit weight and Cost weight. The maximum and minimum values of the weights are therefore also the same (0.8 and 0.2 respectively). The combined effect of the Benefit weight and Cost weight on the Cost-Benefit part of the priority calculation thus ranges from 0.25 to 4.

Type factor. The Type factor in formula 9 in Table 3 is based on the user specified node parameter Type and the user specified project parameters Enhancement weight, Defect weight, NCR weight and Task weight. The Type factor includes the weighting between different node types, and behaves similar to a weight for the whole priority calculation, as seen in formulas 1 and 9. Therefore no additional weight for the Type factor was added. The node parameter Type determines which of the project parameters to use as the value T in formula 9.

$$T \in \{P_{Enhancement\ weight}, P_{Defect\ weight}, P_{NCR\ weight}, P_{Task\ weight}\}.$$

The available choices and their numerical values for the project parameters are the same as those presented for Urgency and CB weights. However the parameters do not have the same default choices. The default choice for each project parameter is presented below.

• Enhancement weight: Medium (0.5), • Defect weight: High (0.65),
• NCR weight: High (0.65), • Task weight: Medium (0.5)

This means that, by default, defects and NCR's will get slightly higher priorities than enhancements and tasks.

The constant *c* in the *Type* factor calculation in formula 9 ensures that the resulting value is close to 1. This is important because it is used to slightly adjust the Priority value. The value of c was determined using the formula 11 in Table 3. From the formula can be deducted that the lowest possible Type factor value is 0.827 while the highest possible value is 1.321.

5 Evaluation

We performed an evaluation of our method. Aspects that were evaluated are Time consumption, Ease-of-use, Expected accuracy, Perceived accuracy, Scalability, Suitability Overall best method. The evaluation consisted in asking six employees at Flygprestanda to prioritize a set of items chosen from a real software project. We choose items for which we expected high medium and low values of prioritization to be represented.

The prioritization was made with 3 different methods in order to compare MAAD prioritization with the method currently used at Flygprestanda (which is ad-hoc, as described in section 3) and with the Wiegers method. We have chosen to compare MAAD with the Wiegers method since we believe it would be Flygprestanda's only alternative to MAAD. The other methods shown in Table 1 are not suitable since they do not consider several factors.

The participants were chosen in order to have different project roles represented: two customers, one manager and three developers. Two of the authors performed a pilot test of the study.

The instruments used in the study were three paper forms used to prioritize the set of project items, one form for each prioritization method. The forms contained item name, description, deadline, and prioritization factors for each item in the set. Two questionnaires were used to evaluate the aspects mentioned above. Except for the time consumption that was estimated in minutes, the other factors were estimated on a 5-point Likert scale, where 1 corresponds to the text "I strongly disagree" and 5 corresponds to the text "I strongly agree". We also interviewed the participants in order to get subjective opinion and experience on using the different prioritization methods, focusing on the use of different factors, scales, and expected impact on Flygprestanda. The answers were annotated using pen and paper.

All the forms, questionnaires and questions used during the semi-structured interviews can be found in [1].

The participants were not told which method was developed by the authors and the methods were referred to as methods 1, 2 and 3. After the prioritization was performed, the participants were asked to fill in the questionnaires. The interviews were performed later once the results of the prioritization had been calculated.

Table 4. Aspects measured during the evaluation with questionnaires.

Aspects measured	MAAD	Wiegers method	Traditional
Time consumption	5 min 25 sec	7 min 35 sec	7 min 45 sec
Ease-of-use	4.33	2.50	3.00
Expected accuracy	4.00	3.50	3.00
Perceived accuracy	3.83	3.00	3.67
Scalability	4.33	3.83	1.33
Suitability	4.00	3.33	2.00
Overall best	5 particip.	0 particip.	1 particip.

Table 4 shows the values of time consumption in minutes for the three methods. The values of the other aspects evaluated are mean values of a normalized Likert scale. The evaluation showed that MAAD prioritization had the lowest time consumption, was the easiest to use, and produced the most accurate results. The evaluation also showed that MAAD prioritization was the most scalable of the methods evaluated. Most of the participants in the evaluation considered MAAD prioritization to be the overall best, and the prioritization method most suitable for Flygprestanda. The results of the questionnaires were also confirmed during the interviews.

6 Conclusions and Future Works

In this paper we have presented the MAAD prioritization method that was developed to support agile, automated and distributed prioritization of requirements, change requests and work tasks at Flygprestanda AB. The method has been implemented as an algorithm in the Scope system and the algorithm calculates priorities automatically. Several factors (Urgency, Benefit, Cost and Type) influence the priority calculation and their relative importance is adjustable. The algorithm can receive input by several stakeholders, both customers and employees at Flygprestanda. An example of the use of MAAD is presented in [1]. An evaluation was also performed showing that MAAD was the best method among those evaluated.

The positive results obtained suggest that the limited number of factors improved both ease-of-use and time consumption of the method as can be seen in table 4 (time consumption). The automated urgency calculation minimizes the risk of missing deadlines, something that several participants to the interview study recognized as highly important to increase product and service quality.

Another strength of the MAAD method is that prioritization does not have to be performed on a set of items, which is the case in methods such as AHP, CBRank, the Hundred-dollar-test and Wiegers method. MAAD makes it possible to calculate the priority of a single item at any time. It also makes possible to compare the calculated priorities of any items. Furthermore, new requirements can easily be prioritized individually, without the need to compare (as in AHP) each newly added requirements with all others which are already present.

However, there are several issues regarding the MAAD prioritization method that could be further investigated. For instance there could be dependencies among items that can affect priorities. Tasks Risk is also a factor that currently is not considered in MAAD prioritization. Including a risk factor would negatively affect both time consumption and ease-of-use but possibly increase the quality of the results. Further investigation could be done on whether a risk factor would increase quality.

Uncertainty is a factor that is not included in the priority calculation in MAAD. An indication of uncertainty regarding effort estimations is however available in Scope since all estimations are visible. Uncertainty should probably not directly affect the priority of an item, but it could be further visualized in Scope by automatically triggering a warning if the variance of effort estimations is high. This would not have any significant negative effects on time consumption or ease-of-use but could possibly lead to increased quality.

Items that do not have specified deadline or target dates are usually wanted as soon as possible, i.e. the sooner they are completed the better. This could call for automatically increasing the priority of "old" items in order to prevent these from being "forgotten". This is achievable without any negative effects on time consumption and ease-of-use. It is however risky since it could possibly result in old, originally low-priority, items to be prioritized over new items that actually are more important. A thorough investigation should be performed before anything like this is implemented in order to assure that it does not cause quality to decrease instead of increase.

A way to specify and visualize the progress of an item is another open issue. This could possibly have a positive effect on urgency calculation and the quality of the results. It could also improve the possibility to track progress of projects. On the other hand it would increase time consumption and decrease ease-of-use as it would require users to frequently update node parameters.

Two factors, Urgency and Cost, are currently based on the mean value of the effort estimations. This could be extended by including a weight for estimators, in order to take into account their estimation accuracy. The estimation accuracy of employees is however hard to determine and it is questionable if this way of combining effort estimations would increase the quality of the priority result.

The impact of the MAAD prioritization method on product and service quality is something that can be evaluated further at Flygprestanda as the use of the Scope system within the company increases. At the time of writing, only few employees at the company use Scope (and therefore MAAD).

Our method has been constructed for the company Flygprestanda and therefore has local validity. Besides investigating further the factors mention above, we plan to test this method in other companies of different size and domain. This will further enhance the validity of our method.

Acknowledgments

We would like to thank the interviewees at Flygprestanda for their dedicated participation in this study, and reviewers of the paper for their valuable comments.

References

1. Gruber, H., and Nae, A., MAAD prioritization A method for agile, automated and distributed prioritization of requirements and work tasks, Master Thesis, Lund University, ISSN 1650-2884, http://sam.cs.lth.se/ExjobGetFile?id=293 (June 2010)
2. Berander, P., Evolving Prioritization for Software Product Management. Doctoral Dissertation Series No. 2007:07, Blekinge Institute of Technology, (2007)
3. Wiegers, K., First Things First: Prioritizing Requirements, Software Development, vol. 7, no. 9 (1999)
4. Karlsson, L., Thelin, T., Regnell, B., Berander, P. and Wohlin, C., Pair-wise comparisons versus planning game partitioning - experiments on requirements prioritisation techniques, in Proc. 8th Int'l Conf. on Empirical Assessment in Software Eng., Edinburgh, pp. 145--154 (2007)
5. Fenton, N.E. and Pfleeger, S.L., Software Metrics: A Rigorous and Practical Approach, 2nd ed. Boston, USA: PWS Publishing Co. (1998)
6. Karlsson, J., and Ryan, K., A cost-value approach for prioritizing requirements, IEEE Software, vol. 14, no. 5, pp. 67-74 (1997)
7. Perini, A., Ricca, F., and Susi, A., Tool-supported requirements prioritization: Comparing the AHP and CBRank methods, Information and Software Technology, vol. 51, no. 6, pp. 1021--1032 (2009)
8. Avesani, P., Bazzanella, C., Perini, A., and Susi, A. Facing scalability issues in requirements prioritization with machine learning techniques, in 13th IEEE International Conference on Requirements Engineering, pp. 297-305 (2005)
9. Avesani, P., Bazzanella, C., Perini, A., and Susi, Supporting the requirements prioritization process - A machine learning approach, in Proceedings of 16th International Conference on Software Engineering and Knowledge Engineering, Banff, Alberta, Canada, pp. 306--311 (2004)
10. Karlsson, J., Wohlin, C., and Regnell, B., An evaluation of methods for prioritizing software requirements, Information & Software Technology, vol. 39, pp. 939-948 (1998)
11. Leffingwell, D., and Widrig, D., Managing Software requirements: A Use Case Approach, 2nd ed. Boston, Addison-Wesley (2003)
12. Regnell, B., Host, M., Natt och Dag, J., Beremark, P., and Hjelm, T., An Industrial Case Study on Distributed Prioritisation in Market-Driven Requirements Engineering for Packaged Software, Requirements Engineering, vol. 6, no. 1, pp. 51-62 (2001)
13. Beck, K., Extreme Programming explained, 2nd ed. USA: Addison-Wesley (2004)
14. Aho, A.V., Hopcroft, J.E., Ullman, J.D., Data Structures and Algorithms. Addison-Wesley, Reading, MA (1983)
15. Clegg, D., and Barker, R., Case Method Fast-Track: A Rad Approach. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1994)
16. Saaty, T.L., The Analytical Hierarchy Process: Planning, Priority Setting, Resource Allocation (New York: McGraw-Hill), (1980)

Prioritizing business process chains for IT optimization

Norman Riegel[1], Oezguer Uenalan[1] and Thomas Jeswein[1]

[1]Information System Development Department
Fraunhofer IESE
Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany
{norman.riegel, oezguer.uenalan, thomas.jeswein}@iese.fraunhofer.de

Abstract. In the present economy, where financial resources have become scarce, the interest in optimizing the interactions between public administration and business is greater than ever - on both sides of the public-private fence. A conscious effort has to be made in order to select those processes and services that promise to yield the highest return on investment in terms of efficiency gains, tax revenues or customer satisfaction, for example. Therefore it is paramount for both sides to identify only those business process chains (BPCs) between public administration and business that are most beneficial for both parties in regard to IT-supported optimization. In this paper, we introduce a method which employs prioritization to streamline this process of identifying precisely those BPCs that are most promising for IT optimization and will truly exhibit measurable gains on both the side of public administration and business.
Keywords: Requirements prioritization, requirements prioritization method, business process, business process chain, information system, business process-driven requirements engineering, E-Government, B2G, G2B

1 Introduction

Business organizations constantly strive to improve their internal business processes to become more successful [4]. However, it is not enough to overhaul and align just the internal procedures and routines. Besides business-to-business (B2B) interactions, it is also crucial to look closely at business process chains (BPCs) between business and public administration (called government-to-business (G2B), or business-to-government (B2G)) and to find ways to improve these interactions. The term *business process chain* (between business and public administration) describes the relative length and higher complexity of integrated processes when several business and governmental stakeholders work together to achieve a common goal, disregarding the limitations of their respective organizations. The foundations of BPCs between business and public administration are legal principles that define the business' information obligation concerning the public, respectively the public administration. A BPC therefore describes the functions needed to fulfil legal obligations in order to inform the acting stakeholders from business and public administration [3].

These relations tend to be bound by strict regulations, standards and laws. But they are not set in stone. Especially since the advent of the Internet and the pervasiveness of web technologies, more and more public administrations feel the need to change procedures and thereby improve services for their customers, i.e., for citizens and businesses. However, since financial resources are scarce, a conscious effort must be made to target for improvement only those processes and services that promise to yield the highest return on investment (ROI), for example in terms of efficiency gains, tax revenues or customer satisfaction [6]. It follows that the potential value of changing any BPC must be determined relative to other BPCs. In other words, the task of the prioritization method is to assess the benefit of undertaking IT-supported optimization of B2G (respectively G2B) BPCs for both business and public administration and thus to create a ranking order for BPCs. A method is clearly needed that provides guidance for the stakeholders regarding the prioritization and the integration of the interests of both stakeholder groups. In this paper, we describe a generic prioritization method which addresses these issues. The approach described here addresses all levels of public administration as instigators of administrative rules, issues, and orders, but also companies which are affected by the aforementioned regulatory system. By using this approach, those BPCs are identified, whose IT optimization is anticipated to be most beneficial for both sides. The paper is structured as follows: Chapter 2 gives an overview of the methodological background, which resulted in the approach presented in this paper. Chapter 3 presents our approach on how to prioritize BPCs between business and public administration. Chapter 4 summarizes and provides an outlook on future work.

2 Methodological Background

In 2005 the state government of Rhineland-Palatinate in Germany commissioned a study to determine the BPCs between government agencies and enterprises in certain industries (automotive, chemical, agriculture) in order to demonstrate the high potential for optimization by means of IT, especially web technologies. With respect to our approach described in this paper, there was one relevant observation during this study: enterprises too often do not see or understand the full potential of "E-Government", i.e. they miss the opportunity to optimize the interconnected links between themselves and public administration. The main reason is a lack of a formal and concerted approach regarding the identification and optimization of BPCs by means of IT [5].

Based on this finding, we therefore developed a preliminary version of the prioritization approach, which was then applied in a feasibility study commissioned by the German Federal Ministry of the Interior. About 200 information and reporting obligations for employers were analyzed and prioritized. The survey among enterprises yielded 17 reporting obligations considered to be most valuable for IT optimization. These 17 reporting obligations were further discussed with five receiving government agencies. Finally, a stakeholder workshop was organized including all involved parties (enterprises and government agencies) as well as additional stakeholders, e.g., expert groups. Its outcome was the identification of the three topmost reporting obligations as a starting point for IT optimization.

This approach proved very valuable since it helped to uncover hidden assumptions on both sides of the fence about the interconnected BPCs and to determine the types of services and processes that should be optimized based on a balanced and objective method [3].

Based on the experience gathered in these two previous studies, we reengineered our methodical approach for the prioritization of BPCs, which is described in the following chapter.

3 The Prioritization Approach

The BPC prioritization basically involves five steps as shown in Figure 1. As input for this approach, a set of preselected BPCs (determined by the concrete project context) to be in focus of the prioritization is taken.

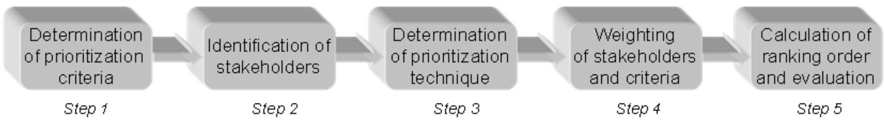


Fig. 1. Steps of the BPC prioritization approach.

Step 1: Determination of quantitative and qualitative prioritization criteria.

Objective criteria (e.g., number of cases) and subjective criteria (e.g., political relevance) are needed to support the prioritization of the BPCs, i.e. to assess the BPCs according to different criteria. Generally, it is hard to find meaningful criteria which serve as basis for prioritization. In our previous studies we already collected a (still incomplete) set of over 20 criteria and created a catalogue for this step of the method (see example in Table 1). The criteria catalogue consists of a description of each criterion, hints on how to elicit or measure the values for this criterion (for objective criteria), as well as some instructions on how further criteria can be collected, for example via empirical studies. As our approach is generic, one has to decide first which criteria are relevant for the concrete project. For example, if BPCs are of interest, where the number of cases are relevant, this criteria will be included in the list of criteria to be prioritized. The selection of criteria should be aligned with the identification of stakeholders in step 2. Only stakeholders that have knowledge about the chosen criteria should participate in the prioritization.

Table 1. Excerpt from the criteria catalogue.

Criterion	Description	Unit	Elicitation method
Frequency	The regularity by which the compulsory registration has to be reported (e.g. quarterly)	1/t	Determination by legal duties or past data

Step 2: Identification of stakeholders.

For the actual prioritization procedure, relevant stakeholders have to be identified who will perform the prioritization according to the selected criteria. Stakeholders can be entire organizations or individual persons. Basically, in the field of BPCs, one can differentiate between business stakeholders and public administration stakeholders. Generally, only companies and public administrations will conduct the prioritization, which are involved in the preselected BPCs. Important stakeholders on the business side are, for example, process owners of the BPC, functional employees who are involved in the BPC, technical staff, and roles responsible for security and legal conformance. Process owners generally are not directly involved in the creation process of a compulsory registration (at best by approval steps), but they are good contact persons to identify other important roles in the organization, which should be involved in the prioritization also.

Step 3: Determination of prioritization technique.

For prioritizing the BPCs, an appropriate prioritization technique has to be chosen, which describes a guideline on how to assign values to the chosen criteria and how to calculate the actual ranking order of the BPCs. In general, the prioritization techniques are independent from the chosen stakeholders, but as mentioned before, it should be ensured that the stakeholders are able to assess the criteria used in the prioritization technique. In the literature, there already exist numerous approaches for the prioritization of alternatives (e.g., [1], [2], [7]). They differ in the way data for individual criteria are determined (different scales, assignment of points, etc.), in the way of the procedure (flat vs. hierarchic), or in the degree of complexity of the procedure. As the techniques are generally not directly applicable for the prioritization of BPCs, we created a collection of state-of-the-art prioritization techniques for our approach, showing advantages and disadvantages as well as modifications that are necessary for computing a ranking order for BPCs. We decided to do this, because different project situations require different techniques that are appropriate. For example, if many BPCs have to be prioritized, some techniques are in favor of others (e.g. through differences in execution time or number of steps in the technique). Also, if tool support is available, some more sophisticated techniques can be used. In the context of BPCs it has also to be regarded for example, that objective and subjective criteria have to be combined in the calculation. Objective criteria might be measured only once for a BPC, while subjective criteria are rated by all stakeholders. Furthermore, certain criteria might only be relevant for one stakeholder group (i.e. assessed by this group). An example of the collection is shown in Table 2.

Table 2. Excerpt from the prioritization technique collection.

Technique	Description	Instructions for use	Necessary extensions/adaptions
Likert scale method (cp. e.g. [7])	The Likert scale method is a simple prioritization technique. It bases on a bipolar scale, where the alternatives are assessed along one aspect. The scale is often used in market research and is an established means for data collection. …	The Likert scale method is a good and simple means for prioritization. However, differentiation is limited, which can be disadvantage in for precise assessments. …	The Likert scale method does not prescribe how to calculate the result of the prioritization itself. After each criteria for each BPC is assessed by each stakeholder, a summation has to be done (including weights), which leads to an overall result for each alternative. …

Step 4: Weighting of stakeholders and criteria.

Where required, stakeholders and criteria have to be weighted, i.e., assigned a weight. The emphasis thus determines the influence of a certain stakeholder or criterion on the overall result of the prioritization. Therefore, a certain value is assigned to every stakeholder, which is included in the actual prioritization. In this context, the stakeholder‘s influence on the project and his/her importance for the project should be considered. Subsequently, the prioritization criteria determined before are weighted. Most often, weighting is needed to point out the importance of a certain criterion. This step is also dependent on step 3 of the method. Depending on the selected prioritization technique, the value assignment takes place in another mode (relative, absolute, etc.).

Step 5: Calculation of ranking order and evaluation of result.

By means of the concrete approach of the prioritization technique, the values for the criteria that have to be evaluated are determined. On the one hand, this is done by measuring (quantitative evaluation); on the other hand, it is done directly through the stakeholders’ judgments. Finally, the prioritization result is calculated, resulting in a ranking order of the BPCs. The ranking order of the BPCs shows the view of the companies and public administrations directly involved. The result is analyzed and interpreted by means of evaluation and comparison by different (groups of) stakeholders or criteria. This means that the results from the enterprises and the public administration are first regarded separately and are combined later. From this, conclusions regarding potential conflicts of interests can be drawn, which can be discussed afterwards. Primarily the actual ranking order is the main result of the prioritization. However, part results of the prioritization could also be interesting. For example, the analysis and separate description of single criteria could give better insights how the result was composed. Also, a sensitivity analysis could be conducted to check the stability of the result of the prioritization, showing how changes to the weights of stakeholders or criteria would influence it.

In order to get an impression that is as broad and valid as possible, the prioritization results have to be verified, negotiated and adapted, if necessary, by involving additional experts and collaterally involved stakeholders (e.g., intermediates and special interest groups). This may be done by means of individual interviews or in workshops.

4 Conclusion and Future Work

In this paper we have presented a generic approach to prioritize BPCs between business and public administration in order to select the best processes for IT optimization in terms of an E-Government roadmap. The approach is based on our previous work in [3] and [5]. It regards criteria relevant for prioritizing BPCs, as well as different stakeholders from business and public administration side. In our future work, we plan to extend our criteria catalogue, to describe how to extend the approach by empiricism and to extend it by experience gained from practice. Furthermore, we want to refine and formalize the negotiation activities in greater detail within the last step of our method. We also want to analyze, how we can align the approach presented in this paper with our prioritization approach described in [4].

Acknowledgements

This work was supported by the project “Pilotierung und Realisierung eines Prozess-Daten-Beschleunigers (PDB) für den Datenaustausch zwischen Wirtschaft und Verwaltung” (Order: B2.28-0004/09/001), funded by the German Federal Ministry of the Interior (BMI).

References

1. 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)
2. Saaty, T. L.: The Analytic Hierarchy Process. McGraw-Hill, New York (1980)
3. Bundesministerium des Innern (ed.): Machbarkeitsstudie zum Forschungsauftrag „Entwicklung von Prozessketten zwischen Wirtschaft und Verwaltung“ Los 3 „Informations- und Meldepflichten für Arbeitgeber“, Berlin (2009)
4. Riegel, N., Adam, S., Uenalan, O.: Integrating Prioritization into Business Process-driven Requirements Engineering. In: 16th International Working Conference on Requirements Engineering: Foundation for Software Quality. Proceedings of the Workshops CreaRE, PLREQ, RePriCo and RESC, pp.113-118. Essen (2010)
5. Steffens, P., Grützner, I., Horch, J., Hufen, A., Jeswein, T, Thomas, L.: Branchenprozesse mit Schnittstelle zur Landesverwaltung Rheinland-Pfalz: Projektabschlussbericht. IESE-Report; 004.09/D, Kaiserslautern (2009)
6. Rombach, D., Steffens, P.: E-Government. In: Nof, S. Y. (eds.) Springer Handbook of Automation, Part I. pp. 1629-1643. Springer, Berlin (2009)
7. Wiegers, K. E.: First Things First: Prioritizing Requirements. Software Development 7, vol. 9, pp. 48--53 (1999)

5 Workshop on Empirical Research in Requirements Engineering: Challenges and Solutions (EPICAL)

Editors

Maya Daneva
University of Twente, Netherlands, m.daneva@utwente.nl

Andrea Herrmann
Axivion GmbH, Germany, AndreaHerrmann3@gmx.de

Technical Programme

EPICAL 2011: Workshop on Empirical Research in Requirements Engineering: Challenges and Solutions

Maya Daneva[1], Andrea Herrmann[2]

[1]M.Daneva@utwente.nl
[2]AndreaHerrmann3@gmx.de

1 Technical Program

The EPICAL workshop took place as a half-day workshop on the 31 March 2011 and gathered 14 attendees. The workshop´s program featured:

- *Kurt Schneider*: Empirical Methods in RE: Purposes and Pitfalls (Key Note Presentation)
- *Marcus Keutel, Werner Mellis*: An In-depth Interpretive Case Study in IS Requirements Engineering Research: Experiences and Recommendations (Paper presentation)
- Expert Panel with the Panelists:
  - *Joerg Doerr*, Fraunhofer IESE and University of Kaiserslautern, Germany, and
  - *Nazim Madhavji*, University of Western Ontario, Canada.

2 Motivation and Goals

Collective efforts of requirements engineering (RE) practitioners, consultants and researchers have yielded a huge variety of solutions for improving requirements processes and artifacts. While it is generally known that the suitability and effectiveness of most of these solutions is contingent to the context in which they are applied, the body of empirical studies that investigate which RE technique is better for which context, is relatively small (Cheng & Attlee, 2007). With few exceptions, little has been done to systematically aggregate the empirical evidence that can possibly confirm or disconfirm the claims of effectiveness of different commercially viable RE approaches that solve particular RE process-related or, product-related, problems. The RE community acknowledges that carrying out empirical research in RE is hard and even, harder compared to other software engineering sub-disciplines, as RE resides in the problem space, while the other sub-fields are focused on the solution space. This workshop called for the explicit discussion on the challenges in setting up good quality RE research designs and promotes the position that for RE research to yield empirically grounded claims, RE approaches need to be systematically assessed by using empirical research methods, e.g. case studies, experiments, action research.

The primary goal of this workshop is to create a forum and a community to debate the need for, the value of, and the challenges in using empirical approaches to researching aspects of RE processes and products. The long-term targeted outcomes are a preliminary agenda for conducting empirical research in RE, and a plan for establishing a forum for exchange of ideas, research designs and research results within the RE community.

We invite readers to review our web site for further information: https://sites.google.com/site/epical2010/

3 Targeted Audience

EPICAL's long term vision is to bring together practitioners and researchers to debate on the research methods suitable in RE, the criteria for judging RE research outcomes, and the implications of choosing particular research designs for the validity of the obtained results. The workshop organizers are committed to provide opportunities for RE practitioners to learn about how to judge the trustworthiness of the current results of RE evaluation research and how to evaluate RE methods themselves. To researchers, the workshop provides a forum to discuss ideas on how to prepare, execute and interpret empirical studies about the effectiveness of RE approaches, and how to generalize from evaluation studies about a specific approach. The workshop is highly interactive in nature.

4 Program Committee

Dan Berry	University of Waterloo, Canada
Jane Cleland-Huang	DePaul University, USA
Daniela Damian	University of Victoria, Canada
Joerg Doerr	Fraunhofer IESE & University of Kaiserslautern, Germany
Olly Gotel	Independent Researcher, New York City, USA
Mahmood Niazi	Keele University, UK
Barbara Paech	University Heidelberg, Germany
Oscar Pastor	University of Valencia, Spain
Bjoern Regnell	Lund University, Sweden
Camille Salinesi	Université Paris 1 Panthéon Sorbonne
Pete Sawyer	Lancaster University, United Kingdom
Kurt Schneider	Leibniz University Hannover, Germany
Inge van der Weer	University of Utrecht, Netherlands
Roel Wieringa	University Twente, Netherlands

5 Key Note Presentation: Empirical Methods in RE - Purposes and Pitfalls, by Kurt Schneider

This key note talk offers a reflection of the following questions that are confronting empirical RE researchers in their studies:

- What are the objectives and results of empirical research methods – especially in requirements engineering?
- What are the pitfalls of empirical research?
- What happens if empirical research is not done?
- What are Best Practices for empirical research?

The talk recommends a number of good practices that researchers could possibly integrate in their research designs and execution plans for their studies:

- Clearly define the purpose of your research effort. Can you achieve it?
- Use threats to validity to guide your experiment design.
- Plan down to the measurement sheet.
- Be prepared for the unexpected.
- Start with a broad claim and add deep details.
- Argue and extrapolate modestly.
- Know your limits. Do not lie!
- Interviews are cheap and weak. What people say they do is not always what they really do.
- Rigour versus relevance: Don´t waste time.
- Document your expectations, e.g. using the Goal-Question-Metrics method.
- Don´t be afraid of empirical research. It is fun, too!

6 Expert Panel

The EPICAL workshop featured a panel with two experts:

- Joerg Doerr, Fraunhofer IESE and University of Kaiserslautern, Germany
- Nazim Madhavji, University of Western Ontario, Canada

The panel included two parts. First, short presentations of the positions of the participating experts regarding the following two questions:

1. What makes empirical research in RE difficult and different from other empirical studies? What turns the replication difficult?
2. How to generalize from evaluation studies in RE?

Joerg Doerr claimed that carrying out empirical research for RE is not more difficult than empirical research in other software engineering fields. One challenge for research is that RE involves both many technical and non-technical roles. Furthermore, RE is domain-specific and domain knowledge is critical in doing good RE work. Before executing RE research, one should identify those context factors that really matter – among the many which are there in practice. In the RE community, experiments have become a prominent RE research technique. The main advantage of student experiments is that the students are available and motivated. However, as RE in practice is not done by novices, but by experts, results of student experiments are not valid for practice. This threat is unique to RE, as opposed to some other downstream software engineering phases, for example coding (where a final year master student would behave as a first year software programmer). Experiments should involve senior practitioners. And empirical papers (and events) should be promoted at conferences.

Nazim Madhavji sees the following challenges in empirical RE research: There are only few RE metrics. RE is much focused on modelling and tool development, less on empirical work. RE is human/ stakeholder-centered, and therefore necessitates the application of research techniques from the areas of social-sciences and qualitative research. Measurement tools and guidelines for questionnaires from social sciences should be used where appropriate. Furthermore, a critical mass of researchers and role models are scarce. Lack of time from RE practitioners also is a problem for RE empiricists. Requirements from real-life projects are confidential, so they may not be publishable and it is also difficult for researchers to get access to them. Student projects are not well generalizable, though they are useful as exploratory studies and for initial insights. What makes replication generally difficult is that RE projects are always different from context to context and the research method must be adapted to the context of concern. Evaluation studies in RE can be generalized by showing reliability by replication and by logical induction.

During the discussion, the panellists and the audience converged on the following points:

- Meta-analysis studies on primary case studies in RE are hard, if impossible, to do. This is because there are very few studies in most of the RE sub-areas, and whatever studies are available they are difficult to compare. Moreover, in case study descriptions, not enough context parameters are given and this further impedes such meta-analyses.
- For a RE researcher to be able to find and engage interested and committed practitioners in an empirical research effort, he/she should be prepared to first invest some time and resources in marketing the RE technique that will be subjected to empirical research. Marketing for RE methods is the precondition for empirical research as empirical research needs reference customers.
- Companies are reluctant to adopt university-conceived innovative tools and tool-supported methods, because in most cases these tools are created by graduate students and PhD researchers and the tool-related expertise walks away once their contracts are over. Senior researchers, e.g. professors, rarely keep updating tools themselves, thus leaving a business partner without any support in the long run (should this partner decide for tool adoption).
- The RE community has probably accumulated a number of empirical studies that might have gone unpublished for a variety of reasons, yet these could well serve the purpose of learning and also the purpose of indicating mechanisms existing behind RE phenomena. Empirical studies are unevenly distributed across application domains. For example, there are more studies for systems that have a relatively longer development history such as large

business information systems (e.g. ERP) for which requirements modelling techniques have been existing for more than 20 years.

- Empirical research in RE contributes in two related but different ways to the body of RE knowledge: empirical studies either explore a RE phenomenon or confirm hypothesised relationships in the area of study. For example, experiments typically serve confirmatory purposes and help learn which technique is better in which context, while qualitative case study are suitable to explore RE problems and possible solution options, and help make implicit knowledge explicit.
- Experiments indicate the presence or the absence of relationships among variables that describe RE phenomena. However, to understand why a relationship is present or absent, more qualitative research studies would be helpful. For example case studies that use in-depth interview techniques or focus groups.
- Qualitative research in RE yields conclusions that are bounded by the data. The researcher does not expect 'absolute truth' and 'absolute generalizability'. A discussion on generalizability should be based on searching for those context characteristics that make comparable the settings in which the qualitative research took place with other similar but different settings. For example, if a RE phenomenon has been researched in a small organization, it makes sense to reason about whether it's logical to observe the conclusions in other small organizations where certain (organizational) mechanisms are in place and others not.

7 Acknowledgement

We thank our PC members, our key note speaker, and our panelists for their participation. Their support and insightful ideas were instrumental to the success of this workshop.

An In-Depth Interpretive Case Study in Requirements Engineering Research: Experiences and Recommendations

Marcus Keutel, Werner Mellis

University of Cologne,
Department of Information Systems and Systems Development,
Pohligstr. 1,
50969 Cologne, Germany
{keutel, mellis}@wiso.uni-koeln.de

Abstract. This article reports our experiences with an interpretive case study, investigating requirements engineering (RE) in practice. First, we briefly explain why interpretive case study research (CSR) is an adequate methodology to answer current research questions in RE. Then, we provide recommendations of how to conduct interpretive CSR in RE based on our own experiences with an in-depth single case study. These recommendations are threefold: (1) initiation, (2) data elicitation and (3) data analysis. Thereby, we aim to contribute to methodological literature on interpretive CSR. This paper mainly addresses less-experienced RE researchers who can use our recommendations as a rough guideline for their own case studies. However, we believe that even experienced RE researchers and researchers from other IS areas will make use of this article as they can reflect on their own approaches.

Keywords: case study, case study research, qualitative research, interpretive, interpretivism, requirements engineering, information systems.

1 Introduction

Scholars request extensive empirical research in requirements engineering (RE) [1, 2]. More intensive research in real-life settings is explicitly demanded [3]. Accordingly, we apply the case study research (CSR) methodology[1] to answer current RE research questions which we will state below. In this article, we use this example case study to develop recommendations of how to do interpretive CSR.

Mingers [5] noticed a strong increase in the number of interpretive studies in leading information systems (IS) journals. However, we noticed that there is a lack of methodological literature on this topic. We found guidelines concerning CSR in general, e.g. [6], but we missed hands-on guidelines of how to conduct in-depth

[1] The terms “method”, “methodology” and “research strategy” are used alternatively by different scholars. Rather than entering into this debate, we follow Piekkari et al. [4] and call CSR a methodology.

interpretive IS CSR. Thus, Walsham's work was very helpful for us. He addresses the nature of interpretive IS CSR [7] as well as interpretive research in general [8]. Thereby, he explains methods for conducting such research. In this paper, we add to his work, as we report more detailed experiences in carrying out such fieldwork. We describe concrete challenges we faced and how to overcome these. Thus, our primary target audience are less-experienced RE researchers who can use our recommendations as a guideline for their own case studies. But we hope that also experienced RE researchers and researchers from other IS areas will make use of this article as they may reflect their own approaches.

The remainder of this paper is organized as follows. In the subsequent section, we explain the research questions underlying our example case study. Then, we describe our case study's research design. Afterwards, we proceed to describe the experiences in conducting our in-depth interpretive case study. The essential results of this section are recommendations of how to conduct interpretive CSR in RE.

2 Our Case Study's Research Questions

The answers to our case study's research questions (RQ) stated below are not part of this article as (1) data analysis is still ongoing and (2) this article's focus is on recommendations of how to conduct interpretive CSR in RE. Nevertheless, it is important to understand these RQs as our case study's research design (cf. section 3) and the resulting experiences and recommendations (cf. section 4) are based on these.

IS scholars have proposed a variety of requirements engineering techniques. These techniques' suitability and effectiveness depend on the contexts they are applied in [1, 2]. Accordingly, current RE research should try to understand the problems that RE practitioners face in choosing and applying RE techniques in order to solve requirements risks[2]. Therefore, the following research questions should be answered:

RQ1: How are different situations of requirements risks characterised in practice?

RQ2: Which techniques are considered and finally chosen to cope with different situations of requirements risks?

RQ3: Why are these techniques chosen in their respective situational context?

RQ4: How successful are the applied techniques in coping with requirements risks in different situations?

To develop answers for these research questions, we apply CSR which is an adequate methodology for the following reasons. Scholars recommend to apply CSR at exploratory studies [11]. Additionally and according to Yin [12], CSR is suitable to answer research questions of '*how*' (cf. RQ 1 and RQ 4) and '*why*' (cf. RQ 3). Finally, we follow Mathiassen et al.'s [2] call for "case studies of the relationship between practices and techniques, of how and why techniques are adopted and combined, and of the effects that techniques have on resolving risks" [2, p. 583]. CSR allows to gain rich, contextual insights into the dynamics of phenomena under investigation [13], in our case the RE practice in coping with requirements risks.

[2] These issues are variously labeled as "risk" or "uncertainty". We follow Mathiassen et al. [2] and use the term "requirements risks". Requirements risks potentially lead to wrong or inadequate software solutions, rework, implementation difficulty or delay [2, 9, 10].

3 Our Case Study's Research Design

The main focus of this article is to give recommendations for doing interpretive CSR in RE. Given the fact that these recommendations are based on our own experiences with such a case study, it is important to describe the underlying research design. This enables the understanding of the anecdotes in section 4. We describe our research design with regard to the following criteria: (1) philosophical foundations, (2) theorizing, (3) case selection, (4) data sources [14] and (5) the researcher's involvement [8].

3.1 Philosophical Foundations

Different philosophical foundations lead to different judgments about the role of CSR, its application, and the criteria for evaluating its quality. Therefore, researchers should clearly state, which philosophical approach they follow [7, 14]. We position ourselves as interpretive researchers. Interpretivism relies on the assumption that people create and associate their own subjective and intersubjective meanings as they interact with the surrounding world [7, 8, 13, 15]. Consequently, interpretive researchers understand the world under investigation and themselves as not separable. Thus, they attempt to understand phenomena by accessing the meanings that participants assign to these. They are aware that their data gathered are their own constructions of other people's constructions of their perceptions of the world. As we will show in the subsequent sections, the interpretive approach has an impact on all other elements of the research design.

3.2 Theorizing

Following Ragin [16], we decided to use case-oriented theorizing. The value of case-oriented approaches is their ability to produce holistic and particularized causal explanations for the outcomes of each investigated case [14]. In this case, theorizing means "tracing the causal processes that generate outcomes in specific contexts." [14, p. 571] Especially the context of a phenomenon under investigation is thus regarded to be very important to derive meaningful explanations. The generalization takes place within a single setting instead of generalizing a theory across different settings [17, 18].

3.3 Case Selection

We select a single software development project (SDP) which we analyze in detail in order to explain its dynamics. Before we started to search for a SDP we established several prerequisites: (1) The requirements for the software to-be-developed should not yet be elicited. This is necessary to become aware of the analysts' perceptions of requirements risks. (2) The project team should have the opportunity to choose RE techniques according to the requirements risks perceived. Dictating the techniques in advance would avert the possibility to analyze the consideration process between different techniques within the SDP. (3) Potential RE techniques should not be excluded because of the geographic distance between customer and contractor. (4) For pragmatic reasons we wanted the SDP to be located in Germany and to be scheduled for a duration not longer than one year.

We chose a strategically important project of a leading international insurance company located in Germany that fulfilled all of our criteria. In section 4.1, the experiences with our search for an adequate SDP are explained in detail.

3.4 Data Sources

Multiple data sources are essential to clarify meaning by identifying different ways a phenomenon is seen [19]. In order to get an in-depth understanding of the investigated SDP, we seek to analyse it based on all available data sources.

During the requirements elicitation phase at least one researcher was on-site every day, participating in meetings, formally and informally interviewing project team members as well as analyzing documents at the project's hard drives. Additionally, we had access to the project's RE management system and the emails of key project members. In our case study, three researchers were involved in data collection on-site.

3.5 Researcher's Involvement

In our case study, we adopt the role as neutral observers. According to Walsham [8, p. 321] neutral means that "the people in the field situation do not perceive the researcher as being aligned with a particular individual or group within the organization, or being concerned with making money as consultants are for example, or having strong prior views of specific people, systems or processes based on previous work in the organization." We extend this definition. In our study, neutral also means to influence the observed phenomena at a minimum as we want to learn from uninfluenced reality. Nevertheless, our continuous on-site presence leads to close involvement, allowing in-depth access to the project and its stakeholders, issues, and data.

4 Experiences and Recommendations of Doing Interpretive CSR

In this section, we present recommendations of how to conduct interpretive CSR in RE, based on our experiences with the research design explained above. We believe that these recommendations are especially helpful for RE researchers new to the CSR methodology. Nevertheless, experienced RE researchers and researchers from other IS areas may find some worthwhile suggestions for their research as well.

We divide this section into three parts: (1) the initiation of our case study, (2) the data elicitation, and (3) the data analysis. However, the three parts of our case study were not performed sequentially. During the initiation, we already collected first data and analyzed the data while continuing the collection on-site.

4.1 Initiation

Search for different partners at the same time. When starting to search for a SDP with certain characteristics (cf. section 3.3), we first focussed on a single company. Given the huge amount of confidential data needed, we participated in many meetings with the company's managers on different hierarchical levels to get their approval for our case study. After deciding to embark on our case study, the company started to search for an adequate SDP. However, it took a couple of months and many meetings with the company's project leaders to realize that we were in an impasse: The company was not able to assign an adequate SDP to our case study. We had lost plenty of time by negotiating with a single company. After this experience, we changed our strategy. We now recommend you to contact a multitude of companies at the same time.

Keep documents simple and practitioner-oriented. When we started to search for a partner and an adequate SDP, we sent out a two-page plain text letter, including our research goals and a request for cooperation. Additionally, we used a 20-slide presentation explaining the research problem, our goals, the methodology, and a detailed current status of our research results. Our intention was to help the recipient develop a comprehensive view on our research project. As a result from our first meetings, we learned that this was too much information. Practitioners are mainly concerned with potential benefits for the company and do not want to be bothered with additional information. Thus, we extremely shortened our presentation. With seven slides, each of them directly addressing issues of our planned case study – especially potential benefits for the company - we kept it simple and practitioner-oriented. This format leads you to much more efficient meetings.

Find a champion. On our way to get access to a SDP, we had to convince a lot of people. In such cases a champion [20] helps to assure the necessary support. It is import to differentiate between a real champion and someone who just pretends to support the case study. At the company which finally participated in our case study, we had strong support by a champion, belonging to the company's middle management. He accompanied us in meetings with project leaders of considered projects. In these meetings, he helped to convince the project members to participate in our case study. The champion explained why our case study must be seen as strategically useful for the company and therefore, the SDP is expected to join our case study. Besides the support in these official meetings, he carefully influenced critical project members in informal conversations and constitutes a positive attitude towards our case study.

Address gut feelings. As we have learned during the initiation of our case study, not everyone can be convinced by factual arguments. Some people follow their gut feelings which you should address. During our initiation meetings we met a lot of project members whom we granted confidentiality and anonymity and explained that our research results will not have any negative consequences for them. Nevertheless, we needed intensive one-to-one conversations to overcome their scepticism. We explained our personal motives in conducting this research project, giving them an opportunity to become acquainted with us and thus convincing them to trust us.

Take more than you need. Our research design demands us to observe just one SDP. Nevertheless, we recommend starting to observe more projects if possible. At our partner company, we initially observed two projects, with the intention to drop the less interesting project after a couple of weeks or months. This turned out to be a good decision: One project showed a lot more potential for our research topic because of more situations dealing with requirements risks. Before you begin to observe a project you do not know what data you will finally get.

Clarify conditions and expectations. Conducting an in-depth case study in a RE context implies having a lot of stakeholders at the company's site, e. g. the company's management, the project leader, software developers, business analysts and customer representatives. Each of them has different expectations concerning the case study's output. When you introduce yourself to a project, you should clearly state, what deliverables your case study will have and how each stakeholder may benefit from them. This helps to strengthen their commitment to the case study and avoids having frustrated stakeholders at the end of your research. Additionally, a clarification of the data elicitation conditions is needed at the beginning of a case study. We made sure that everyone in the project knows what kind of data we are interested in and that it is important for the success of our study to receive all relevant information regarding our research questions. Thus we encouraged the project members to forward us all information possibly relevant for us.

4.2 Data Elicitation

Build trust. People grant access to the information you need only if you are trustworthy. Otherwise, they will tend to hide potentially critical or harmful information. Consequently, we invested plenty of time in networking with project members, e.g., by meeting for lunch or dinner, participating in project team events and informal conversations. These meetings were mainly about non-research related topics. Nevertheless, we did not interrupt our dialog partners when they referred to project related issues. It even occurred that they asked for our opinion regarding other project members or they tried to get information which they assumed we received from another project stakeholder. In these situations, we consequently showed our integrity and confidentiality by neglecting any answer. In most cases, this did not lead to any resentment but to more trustful conversations, containing interesting information regarding our research questions. Nevertheless, it is important to keep a professional distance from each project member. Otherwise, the researcher may become socialized to their specific views and thus may loose the benefit of a fresh outlook on the situation [8].

Collect data broadly. It seems obvious that you should have a clear focus in the data collection within an in-depth case study. In our opinion, limiting the focus too much would be a mistake. During our case study, we participated in a multitude of meetings, which initially just had a peripheral link to our research questions, e.g., effort estimations or conversations about training courses. However, during these meetings issues aroused which directly affected them. Consequently, we recommend using every opportunity to gather data, which may help to answer your research questions. That means for example that the decision about observing a meeting or not should not solely be made based on the planned topic but on the list of scheduled participants, as contents may shift occasionally.

Take notes without attracting attention. It is important to take notes when you are observing a meeting or talking to project members in order to preserve the information. Taking notes just may become a problem, if you do it very conspicuously. In one of our first meetings at the observed SDP, we continuously took notes. This seemed to irritate some project members. Apparently, we influenced them and some started to be afraid of giving critical comments in the meeting. After a while one participant said: "I would like to know, what you are writing down all the time". For future meetings we learned to behave differently: Now we are noting keywords, sometimes delayed, e.g. not directly after critical comments are expressed. The notes are completed after the meeting is finished. With regard to informal conversations, we take notes only afterwards. This is for the same reason – we do not want to scare the project members.

Share impressions with research colleagues. As stated above, three researchers were collecting data on-site simultaneously. We did not divide the data elicitation in different topics due to pragmatic reasons: We avoided the necessity of having each researcher on-site every day. Consequently, different researchers got into contact with the same topics and project members during their on-site presence. If you follow this approach you need an intensive and regularly sharing of impressions between the researchers, mainly because of two reasons: (1) Project members do not like to be bothered by being asked the same questions twice. (2) In order to understand current discussions, the observing researcher has to be up-to-date concerning the state of the SDP.

Carefully involve the champion. In section 4.1 we stated that a champion is very useful for the initiation of a case study. Of course, the champion can also be helpful during data elicitation, but you should involve him very carefully. We rarely involved him, just in case of challenges which had their origin beyond our sphere of action. Thereby, we avoided having project members feeling under pressure because of the champion's presence. Nevertheless, we stayed in close contact with him, e.g. through weekly lunch meetings, in order to assure his support in potential crisis situations.

Remind project team of your presence. In our case study we learned that after a while some people tended to forget us. We expected this behaviour and thus did not rely on actively being informed about every new development by the project team, e.g. in form of scheduled appointments. Consequently, we implemented some counteractive measures: E.g., we assured access to the key project team members' online calendar and checked regularly if there were are any relevant meetings to which we were not invited. In such cases, we asked the meeting's organizer politely if we could participate. Afterwards, we explained again that it is very important for our study's success to get all relevant information regarding our research questions. Usually the effect was that this SDP team member got a bad conscience. He or she then promised to keep us better in mind and in most cases his turned out to be true. In the following weeks, the information flow concerning the affected project team member was much better than before but sometimes after a while experienced another worsening. Therefore, you should remind the project team members of your presence from time to time.

4.3 Data Analysis

Regularly reflect on what you have learned. We followed Walsham's recommendation of preparing sets of themes and issues after a certain period of time of data collection [8], that is after a set of interviews or meetings. This first analysis requires a reflection of previous insights and may also lead to redirections in data elicitation. As interpretive researchers, we are aware of our subjective views on the elicited data (cf. section 3.1). Each researcher involved creates a subjective and independent view of the world under investigation. Consequently, each researcher involved should independently summarize his or her findings from time to time. Consolidating the individual insights leads to a more holistic picture, representing all perceptions and thoughts.

Make use of software tools. Even though we agree on Walsham that "software does not remove the need for thought, as the choice of themes remains the responsibility of the researcher" [8, p. 325], we recommend to make use of software tools. During our case study, we collected a multitude of data. According to our experiences, it is very helpful to use a software tool to support the qualitative data analysis, such as QSR Nvivo, the product we use. Such a product is helpful to arrange the data and facilitates collaborative work with multiple researchers. It is also regarded as helpful to justify your findings with evidence, as it helps to link findings back to the original data that supports it.

Reflect with practitioners. Given our positioning as interpretive researchers, we understand phenomena through accessing the meanings that participants assign to these phenomena (cf. section 3.1). These meanings are not obvious in every case. To avoid being on the wrong track, you should reflect preliminary perceptions and interpretations in informal interviews with practitioners. In some cases, this led us to interesting reinterpretations of observed phenomena, such as motives for choices between different RE techniques.

5 Conclusion and Limitations

We derived this article's recommendations from our own experiences in conducting an interpretive in-depth case study. Due to their nature, these recommendations are subjective. Not every recommendation may turn out to be useful in every setting as they are derived from our specific context. A further limitation is the unfinished status of our case study. There may be some more pitfalls, especially during data analysis, which we are not aware of right now. Nevertheless, we believe that the recommendations stated above will help less-experienced RE researchers when conducting such a case study.

We encourage the research community to debate different ways how to conduct CSR. We believe that concrete experiences concerning the adoption of a research methodology help to further develop the methodology. Therefore, especially experienced researchers are requested to share their knowledge of the methodology-in-use. We hope that our recommendations are also inspiring for those researchers and therefore lead to improvements in their future research projects.

References

1. Cheng, B.H.C., Atlee, J.M.: Research Directions in Requirements Engineering. In: Future on Software Engineering 2007, pp. 285--303. IEEE Press, Washington (2007)
2. Mathiassen, L., Tuunanen, T., Saarinen, T., Rossi, M.: A Contigency Model for Requirements Development. Journal of the Association of Information Systems 8(11), 569--597 (2007)
3. Condori-Fernandez, N., Daneva, M., Sikkel, K., Wieringa, R., Dieste, O., Pastor, O.: A Systematic Mapping Study on Empirical Evaluation of Software Requirements Specifications Techniques. In: 3rd International Symposium on Empirical Software Engineering and Measurement, pp. 502--505. IEEE Press, Washington (2009)
4. Piekkari, R., Plakoyiannaki, E., Welch, C.: 'Good' Case Research in Industrial Marketing: Insights from Research Practice. Industrial Marketing Management 39, 109--117 (2010)
5. Mingers, J.: The Paucity of Multimethod Research: A Review of the Information Systems Literature. Information Systems Journal 13(3), 233--249 (2003)
6. Runeson, P., Höst, M.: Guidelines for Conducting and Reporting Case Study Research in Software Engineering. Empirical Software Engineering 14(2), 131--164 (2009)
7. Walsham, G.: Interpretive Case Studies in IS Research: Nature and Method. European Journal of Information Systems 4(2), 74--81 (1995)
8. Walsham, G.: Doing Interpretive Research. European Journal of Information Systems 15(4), 320--330 (2006)
9. Barki, H., Rivard S., Talbot, J.: An Integrative Contingency Model of Software Project Risk Management. Journal of Management Information Systems 17(4), 37--69 (2001)
10. Lyytinen, K., Mathiassen, L., Ropponen, J.: A Framework for Software Risk Management. Journal of Information Technology 11(4), 275--285 (1996)
11. Eisenhardt, K.M.: Building Theories from Case Study Research. The Academy of Management Review 14(4), 532--550 (1989)
12. Yin, R.K.: Case Study Research: Design and Methods (4th ed.). Sage, Newbury Park (2009)
13. Dyer, W.G., Wilkins, A.L.: Better Stories, Not Better Constructs, to Generate Better Theory: A Rejoinder to Eisenhardt. The Academy of Management Review 16(3), 613--619 (1991)
14. Piekkari, R., Welch, C., Paavilainen, E.: The Case Study as Disciplinary Convention: Evidence From International Business Journals. Organizational Research Methods 12(3), 567--598 (2009)
15. Orlikowski, W.J., Baroudi, J.J.: Studying Information Technology in Organizations: Research Approaches and Assumptions. Information Systems Research 2(1), 1--28 (1991)
16. Ragin, C.C.: Turning the Tables: How Case-Oriented Research Challenges Variable-Oriented Research. Comparative Social Research 16(1), 27--42 (1997)
17. Geertz, C.: The Interpretation of Cultures. Basic Books, New York (1973)
18. Lee, A.S., Baskerville, R.L.: Generalizing Generalizability in Information Systems Research. Information Systems Research 14(3), 221--243 (2003)
19. Stake, R.E.: Qualitative Case Studies. In: Denzin, N.K., Lincoln, Y.S. (eds.) The Sage Handbook of Qualitative Research (3rd ed.), pp. 443--466. Sage, Thousand Oaks (2005)
20. Benjamin, R.I., Levinson, E.: A Framework for Managing IT-Enabled Change. Sloan Management Review 34(4), 23--33 (1993)

Part II

REFSQ 2011 Empirical Track Proceedings

6 Empirical Research Fair

Editors

Nazim H. Madhavji
University of Western Ontario, London, Canada, madhavji@gmail.com

Brian Berenbach
Siemens Corporation, USA, brian.berenbach@siemens.com

Technical Programme

Introduction to the Empirical Research Fair at REFSQ 2011

Nazim H. Madhavji[1] and Brian Berenbach[2]

[1] University of Western Ontario, London, Canada
madhavji@gmail.com
[2] Siemens Corporation, USA
brian.berenbach@siemens.com

Introduction

It is widely recognised in the requirements engineering (RE) community that case studies of industry projects are critical for our in-depth understanding of both: (a) the phenomena occurring in projects, processes, systems, and services and (b) the impact of the phenomena on the quality, cost, and deliverability of systems. For example: Are we doing too much or too little documentation of software and system requirements in projects? When eliciting requirements for a relatively new system or a legacy system, should we examine the existing system architecture and, if so, when, and why then? What is the impact of decisions made during RE on the quality, cost and delivery of software or services to the customer? How do we manage a very large number of requirements on large projects? How does one efficiently and effectively review the requirements for a large system? Such questions abound in industrial projects, but there is a dearth of research results in RE that actually helps solve them. Consequently, opportunities are being lost to make improvements in RE from a number of angles.

This situation is akin to paddling a bicycle with a malfunctioning sprocket. The RE sprocket needs to be fixed so that (i) industry can provide the right kind of questions and environment to the researchers to solve important RE problems and (ii) research in RE can yield relevant answers to help RE practice, thus creating a symbiotic relationship between industry and academia.

For what appears to be the first time ever and certainly for the first time at any REFSQ working conference, at REFSQ 2011, there was a special “Empirical Research Fair” track, the goal of which was to bring together practitioners and researchers together – very much like in a fair – so that: (1) practitioners can propose studies that organisations would like to have conducted and (2) researchers can propose studies they would like to conduct in industry. It is a meeting point to match the demand and supply of empirical studies in RE among researchers and practitioners.

The format of this session was poster displays of the case study proposals by both practitioners and researchers; in essence, a “match-making” session to bring together mutually interested industry personnel and researchers to “hit it off” with possible collaboration on the research topic displayed.

There were twelve proposals in all, which are described following this introduction.

Committee Members

Empirical Fair Program Committee Co-chairs

- Nazim H. Madhavji, University of Western Ontario, Canada
- Brian Berenbach, Siemens Corporation, USA

Program Committee Members

- Joy Beatty, SEILevel Company, Texas, USA
- Jeff Carver, University of Alabama, USA
- Daniela Damian, University of Victoria, Canada
- Remo Ferrari, University of Western Ontario, Canada
- Samuel Fricker, University of Zurich, Switzerland
- Thomas Gehrke, Siemens Rail Automation, Braunschweig, Germany
- Martin Herget, Siemens Corporate Technology, Erlangen, Germany
- Barbara Paech, University of Heidelberg, Germany
- Bjorn Regnell, Lund University, Sweden
- Karen Smiley, ABB Corporation, North Carolina, USA
- Roel J. Wieringa, University of Twente, Netherlands

What Do User Stories Tell Us about the Business Value?

Zornitza Bakalova, Maya Daneva
z.bakalova@utwente.nl, m.daneva@utwente.nl
University of Twente, the Netherlands

Introduction: What we want to study and why this is important?

Although the agile methods are spreading fast in the industry and are getting increasing attention from the research community, the topic of the quality of agile requirements engineering (RE) artifacts – the so-called user stories (USs), seems to be under-researched. Advocates of agile methodologies (AMs) claim that AMs reduce waste by implementing only those requirements that bring value for the customers. For USs to help reduce waste, they should satisfy six quality criteria [Cohn], the most important one being that USs should be *Valuable* (to customers or users). The literature sources on AM recommend that ideally, the value should be stated explicitly on the story card. Our previously published research, however, revealed that business value evades explicit specification: we found that the business value more often than not is present only implicitly or assumed as a tacit knowledge of those involved in agile RE.

Our goal in this empirical study is to get a deeper understanding of how the agile requirements, used in practice, specify business value and how agile project teams members (both customers and developers) reason about customers' value and customers' value-creation. We will analyze the agile requirements artifacts with respect to: (i) format, (ii) content, and (iii) level of granularity of the documented business value statement. The results of the study will help identify and distill good practices for writing useful USs that do support the value creation process. Well written USs with explicit value statements could help to identify unnecessary requirements and thus reduce waste for the customers. Furthermore, they can help for value-driven decisions-making on requirements priorities at inter-iteration time.

Wanted from industry:

We welcome collaboration with any agile organization that uses USs as requirements documents for their projects. Ideally we consider two to four projects of different sizes to be enough for the purpose of our research. We would need a max of 5 person/hours distributed over 4 weeks of time from the representatives of the participating organization. The best fit would be people writing the user stories or prioritizing them. The time will be spent on interviewing the representatives from the company to understand the project context and to share our results. We consider our research to be of value to the company, because it may shed light into the quality levels of the company's agile requirements specifications and, if necessary, it also would suggest improvement actions to increase the quality of the USs with respect to value specification.

Work Plan

Our work plan includes the following: First we will analyze the USs created in different projects. Second, practitioners from the organizations will be interviewed about the practices of the organization with respect to writing the USs and using them further during the development and decision-making process. The interviews will be semi-structured, with a questionnaire prepared up-front. The data will be analyzed by means of the Grounded Theory approach.

RBAC in Practice

Virginia N. L. Franqueira and Nelly Condori Fernandez

University of Twente
Enschede, The Netherlands
{FranqueiraV,N.CondoriFernandez}@ewi.utwente.nl

1 Introduction

Since the Role-Based Access Control (RBAC) model was first introduced [2], it evolved into probably the most discussed and researched access control model in academia. It became the basis for hundreds of textbooks, research prototypes and theoretical studies. Specially after the NIST (National Institute of Standards and Technology) standard for RBAC [4] was officially approved by the American National Standards Institute [1], RBAC features also gained a lot of attention of high profile commercial products. Its basic feature, which decouples the assignment of users to permissions via roles (illustrated in Figure 1), together with additional features are claimed to allow an efficient management of permissions, an effective enforcement of the need-to-know principle, and a scalable assignment of permissions to users. However, it is unknown to what extent the efforts put into RBAC research and development make true their promise in practice. Are they really aligned with the needs of the practitioners? Requirement engineers should be the first to know the answer!

Fig. 1. RBAC basic feature: assignment of users to permissions via roles [3]

In an earlier literature study, we collected: (a) a set of core features of RBAC, (b) its assumptions and strengths, and (c) a set of phenomena which may limit these strengths in practice. This study revealed that roles can be used to control access to information in: *support applications*, with operating system-specific roles; *stand-alone business applications*, with application-specific roles; *enterprise-wide applications*, with roles shared among several applications; and *cross-enterprise applications*, with roles shared among several enterprises. This empirical proposal builds upon our initial study and aims to verify to what extent these features, assumptions, strengths and phenomena are recognized and important in practice, and also aims to complement our knowledge with additional strengths and phenomena, collected from practitioners.

2 Wanted from Industry

To achieve our goal, we are seeking to gain a broad instead of a deep knowledge of RBAC in practice, i.e. we look for a large number of organizations of any size (e.g., small to multinationals) and from any sector (e.g., banking, government, telecom). One professional per organization experienced with *role engineering* ("the process of defining and implementing roles" [5]) and/or *role management* would be ideal. However, we impose no restriction on how this experience has been acquired. Therefore we welcome, e.g., system administrators, consultants, risk managers, information security officers, IT architects, decision makers, Identity and Access Management experts.

3 Work Plan

Our research strategy includes two steps. The first step is an online survey (requires 0.5 hour) to do a quick scan of the use and experience with RBAC across organizations. The second step will build up on the survey and will include one in-depth interview per organization (requires 1.5 hour) to understand reasons behind choices and get any other background information relevant to the use of RBAC. In the end, the participants will receive a summary report of the results which can help them either to improve the use of RBAC in their organization, or can help them to learn if pitfalls of RBAC they experienced in practice are echoed by the experience of other organizations. We expect the whole process will take three months to complete, after the survey is launched.

References

1. ANSI/INCITS_359: Information Technology - Role Based Access Control. American National Standards Institute (ANSI), International Committee for Information Technology Standards (INCITS) (February 2004)
2. Ferraiolo, D.F., Kuhn, D.R.: Role-Based Access Controls. In: Proc. of the 15th NIST-NCSC National Computer Security Conference. pp. 554–563 (October 1992)
3. Ferraiolo, D.F., Kuhn, D.R., Chandramouli, R.: Role-Based Access Control. Artech House, Inc., Norwood, MA, USA (2003), ISBN: 1-58053-370-1
4. Ferraiolo, D.F., Sandhu, R.S., Gavrila, S.I., Kuhn, D.R., Chandramouli, R.: Proposed NIST Standard for Role-Based Access Control. Information and System Security 4(3), 224–274 (2001)
5. Gallaher, M.P., O'Connor, A.C., Kropp, B.: The Economic Impact of Role-Based Access Control. Tech. Rep. RTI Project Number 07007.012, National Institute of Standards and Technology (NIST) (March 2002)

What Do Homecare Provider Stories Tell Us about Dynamicity?

Alireza Zarghami, Mohammad Zarifi Eslami, and Marten van Sinderen

University of Twente, The Netherlands
{a.zarghami,m.zarifi,m.j.vansinderen}@utwente.nl

Introduction:
The growth of the ageing population in industrialized countries is expected to become a serious problem in the near future. A promising solution to this problem is to improve healthcare systems and to support independent living for elderly by way of homecare service provisioning [1]. An important challenge to realize homecare service provisioning is to find effective ways to handle the dynamicity demands of the homecare domain. These demands imply that the provided services must be adapted based on a) frequently occurring contextual changes like a care-receiver's location and occupation, or b) slowly developing changes in a care-receiver's requirements like his/her extent of impairment [2]. To address these changes, the provisioning system should be capable of adapting the homecare services while they are being executed. At the same time, the execution of homecare services should be constrained by a service plan which is created by care-givers possibly through a separate system. The purpose of the service plan is to specify at a high level of abstraction how to address the needs of the care-receiver (in compliance with applicable medical protocols).

The service plan (especially its completeness and accuracy) plays an important role in the success of the provisioning system to deal with the dynamicity. In our definition, a service plan refers to one or more service building blocks (SBBs) and it describes the configuration and orchestration of instances of these SBBs as well as decision rules with respect to run-time behaviour. The SBBs, like a medicine dispenser or reminder, are the smallest manageable services which cannot be broken down further into smaller services from the care-givers point of view. Configuration parameters allow the care-givers to specify different aspects of the SBBs such as service operations and user interface modalities. Orchestration schemes determine how SBBs are composed. Decision rules determine the possible adaptation at runtime, based on evaluation of the rules with runtime data (e.g., context values). For example, decision rules can be used to choose between alternative operations of one SBB or between alternative data and control flows among the SBBs, based on specific runtime circumstance. We believe that enhancing the service plan with decision rules can address the dynamicity in the homecare domain, especially with respect to existing rule-based and careflow medical protocols [3].

Expectation: What input is needed from industry?
To investigate the feasibility of our approach, we need to obtain several homecare scenarios and their possible variations from homecare providers, either for

different care-receivers or for a specific care-receiver with different runtime circumstances. We would like to interview a maximum of three representatives from different participating organizations who provide homecare services. The interviews will last 30 minutes. The interviewees should have at least 3 years experience with homecare services (Remote Patient Monitoring and Treatment) and can be either IT specialists or nurses. During the interview, we present several predefined service plans which have been developed based on some example scenarios and ask the interviewee to match them with the real scenarios which they have faced in practice. Our aim is to find out: (1) what types of SBBs and configuration rules can be useful for homecare services; (2) what types of ICT-based orchestration schemes and composition rules have been employed in practice; (3) what types of changes have been experienced and how often are homecare providers faced with such changes in their homecare service provisioning; (4) what types of medical protocols are imposed by legislation or regulation and how these affect the ICT-based homecare services; and (5) to what extent a separately devised service plan can help without compromising the safety and privacy in the homecare domain.

Work Plan:
Based on the interviews, we will use qualitative data analysis techniques (coding) to extract the common patterns of orchestration schemes, SBBs' configuration parameters and applicable decision rules in order to address dynamicity in homecare, i.e. to enhance adaptivity in homecare service provisioning. Then, we will refine our approach to support the scenarios collected from the interviews. In addition, the SBBs, their configurations and orchestration schemes as well as related decision rules will be deployed in our provisioning system to demonstrate its capability for runtime service provisioning. We also plan to enhance the evolvability of the homecare services based on monitoring performed by the provisioning system to improve off-line modification. This work is part of the IOP GenCom U-Care project which is sponsored by the Dutch Ministry of Economic Affairs under contract IGC0816.

References

1. Ganer, K., Conrad, M.: ICT enabled independent living for elderly, A status-quo analysis on products and the research landscape in the field of Ambient Assisted Living (AAL) in EU-27. prepared by VDI/VDE Innovation und Technik GmbH (March 2010) Available at: http://www.ehealthnews.eu/publications/latest/2061-ict-enabled-independent-living-for-elderly, last visited: October 2010.
2. McBryan, T., McGee-Lennon, M.R., Gray, P.: An integrated approach to supporting interaction evolution in home care systems. In: PETRA '08: Proceedings of the 1st international conference on PErvasive Technologies Related to Assistive Environments, New York, NY, USA, ACM (2008) 1–8
3. Gomoi, V., Stoicu-Tivadar, V.: A new method in automatic generation of medical protocols using artificial intelligence tools and a data manager. In: Proc. Int Computational Cybernetics and Technical Informatics (ICCC-CONTI) Joint Conf. (2010) 243–246

Interaction With Stakeholders Made Easy
From Intuition to Structured Facilitation

Lauri Ann Scheinholtz[1,2] and Ilona Wilmont[3]

[1] Mithun Training & Consulting, Leusden, The Netherlands,
[2] Laquso, Eindhoven University of Technology, Eindhoven, The Netherlands,
`la.scheinholtz@mithun.nl`,

[3] Institute for Computing and Information Sciences, Radboud University Nijmegen, Nijmegen, The Netherlands,
`i.wilmont@science.ru.nl`

1 Introduction

Communication between domain experts and requirements experts remains a problem, from knowledge elicitation to formal model review. The latter are hard to understand and require a certain, often trained, way of thinking. Yet active stakeholder participation, from user to manager, is vitally important to avoid issues such as misunderstanding, lack of consent over solutions and change resistance.

Practitioners report different experiences when dealing with stakeholders. Some of them participate really well and show interest, others would rather see them return with something they can understand. Why can some people deal with formalisations and structured knowledge elicitation so much better than others? Many practitioners also mention assessing stakeholders before they even come to a session, however they cannot describe explicitly how this assessment proceeds. 'Based on gut feeling' is the most commonly provided answer when asked.

Obviously then, a stakeholder's unique profile influences how he performs in an elicitation or review session. On the other hand, one of the authors' experiences show that the requirements engineer's way of questioning the stakeholder also significantly influences the information that will be elicited. We want to gain more insight into these phenomena, which are currently mainly guided by practitioners' intuition. A more structured description can result in more certainty about which people to include in a session, and how to facilitate the session to match the styles of the people involved.

This leads us to ask the following research questions:

- What is the influence of a stakeholder's way of reasoning, knowledge and skills, and attitude towards the project, on his performance in an elicitation or model review session?
- How do practitioners' interviewing styles influence the amount and usability of information elicited from stakeholders?

– How understandable are formal models for business stakeholders, even the ones that claim to be based on human perceptual principles?

Adapting to the stakeholders perception and way of thinking, which creates a sense of ownership over business problems among the people participating in the actual process, will lead to more efficient business processes and reduced errors and costs in system design.

Eventually we want to develop guidelines for practitioners to help them facilitate sessions and conduct interviews, which are specifically selectable based on the situation and the people involved. As a participant in our research we would like to give you as many benefits as possible. As every interviewer and organization has different interests in research results here are some of the benefits you can receive:

– Early access to research results
– Information on how you compare in method and style to other interviewers
– General strengths and weaknesses based on empirical data
– Deeper insight into communication processes within your interview and possibly your organization

2 Wanted from Industry

To tackle this problem, we would like to observe several aspects of a collaborative session: verbal and nonverbal interaction during the session and the preparation before the session. We look at motivations that lead people to perform certain behaviour, what skills they may have that will aid or inhibit them, their cognitive activities and how team performance changes as time progresses. For this, one researcher sits in on and audio or video-records the session, which will be transcribed and analysed afterwards. On top of that, we would like to assess the participants cognitive styles before the session, and conduct a short survey on experiences and comprehension of the material covered after the session.

We are looking for industry projects which concern system design, business process design or business process analysis. Specific types of sessions can be but are not limited to: requirements elicitation, brown paper sessions, model building, model validation. Facilitated or guided sessions by an analyst or other expert are welcome, as are self-facilitated sessions or sessions using some form of tooling or method.

Requirements Elicitation and Validation for Secure IT Enabling Supply Chain Networks

Elena Irina Neaga[1], Michael Henshaw[1]

[1] Engineering System of Systems (EsoS) Group, Dept of Electrical and Electronic Engineering, Garendon Wing Bldg., Holywell Park, Loughborough University, LE11 3TU Loughborough, United Kingdom
{E.I.Neaga, M.j.d.Henshaw}@lboro.ac.uk

Abstract. This research fair proposal deals with business driven requirements engineering and management for the development of secure software systems that support effective supply chain collaboration. To enable effective implementation of security concepts for enterprises, including their supply chains, it is necessary for software developers to work alongside systems engineers who model the requirements in architecture at the "systems of systems" level. The target of this proposal is identification of case studies for practical approaches in order to capture the dynamic nature of software security requirements to support industrial supply chain networks. The proposal will also identify mechanisms through which software security requirements can be better elicited, expressed, and validated for effective system development as part of an integrated architectural enterprise approach to supply chain networks and solutions for changing requirements that arise as a result of the dynamic nature and behaviour of threats and vulnerabilities.

Keywords: requirements elicitation and validation, secure software systems, effective supply chain collaboration, vulnerabilities.

1 Introductory Aspects

IT enables supply chains to become more efficient and agile through enhanced collaborative working, but this appears to come at the expense of increased risks to information security vulnerabilities. Service oriented architectures have been the significant enablers, but the advent of cloud computing is now leading to even greater cost reductions and this field is expected to grow within the commercial world. This poses new, and complex, challenges for security protection and increases the difficulty for requirements management associated with software security. The rapid increase of the already large number of IT support systems being deployed as intranets and extranets in supply chain collaborations creates new possibilities for significant security vulnerabilities and threats. For instance, financial transactions can be interrupted or misdirected by the hackers or crackers; collaborative supply chain information may increase the risk of revealing sensitive information to competitors; logistics information can be illegally used to disrupt normal transportation operations, and attackers can break into an organization's supply chain infrastructure to disrupt or totally collapse its operations and functions. Whilst these assumptions have not been completely validated by supply chain and security practitioners, it is noted that in a recent UK Government Cabinet Report the annual cost of cyber crime to the UK economy was estimated for the first time to be £27Bn and that theft of intellectual property (IP) accounted for approximately £9Bn (http://www.cabinetoffice.gov.uk/sites/default/files/resources/the-cost-of-cyber-crime-full-report.pdf).

2 Industry driven Approach

Within an industry/Government-sponsored the programme called Integrated Model for Governance, Risk and Compliance (iGRC) (http://www.igrc.co.uk), we have analysed supply chain vulnerabilities due to the use of IT-enabled collaboration and contingency planning. IT security of supply chains requires that the IT services and security software are viewed and developed within the context of the supply chain enterprise as a whole and the benefits of the systems engineering approach to integrating requirements modelling with (software) system analysis, design and development have been demonstrated. Baseline requirements for secure IT solutions to support supply chain collaboration have already been elicited.

2.1 Wanted from Industry

This approach driven by practical needs will advance the requirements engineering process applied to secure IT services through the active involvement of industrial practitioners with knowledge of supply chain management and experience in the specification of IT systems. We seek practitioners willing to help us develop case studies through interviews and/or discussions. We wish to consider the general issue of IT-enabled supply chain vulnerability, but will have a particular focus on organizations working in hi-tech areas in which loss of IP is an area of concern.

A framework for case study definition that focuses on requirements elicitation, verification and validation based on modelling approaches used within the iGRC research programme will be prepared in advance to assist effective information capture. Engagement with industry through the Empirical Research Fair will support validation of the iGRC research undertaken for:

- Approaches to the definition, expression and validation of software security requirements in the context of enterprise supply chain collaboration.
- Definition of real world solutions for changing requirements that arise because of the dynamic nature of threats and vulnerabilities that affect software systems.
- Development of an integrated modelling approach, validated by real world exemplars.
- Case studies of software systems implementations that draw out the challenges associated with management of system security requirements.

Scalability of Methods for Managing Requirements Information – an interview study

Krzysztof Wnuk and Björn Regnell,

Department of Computer Science, Lund University,

{Krzysztof.Wnuk, Bjorn.Regnell}@cs.lth.se

Introduction

Storing, updating and tracing changing large amount of requirements can be challenging and often overwhelming. For storing requirements information MS Word or Excel are, next to requirements databases, widely used tools. However, when the size of the project grows, the need for robust, coherent, extensible and flexible ways of structuring requirements information has been stressed by practitioners**.** The purpose of this exploratory interview study is to *investigate the experiences from using various methods for storing requirements as well as structuring the requirements information*. The second goal of this study is *to reflect on the scalability of used methods for storing requirements* (for example MS Excel or requirements databases) as well as the *scalability of the methods of partitioning requirements information*. Some example questions to be asked during the interviews:

- How do you store and manage requirements in your projects?
- How do you organize the requirements and associated information?
- Are non-functional requirements placed next to corresponding functional requirements or in a separated section/module?
- What is according to you the maximum/minimum amount of requirements that can be stored or managed using the current technique?
- What are the issues related to the current way of storing and managing requirements information?
- Does the current way of sorting and managing information make easy or hinders the following tasks: change impact analysis, scope management, traceability, reuse of requirements, requirements verification and validation and finding obsolete requirements.

Wanted from Industry

Experiences from all kind of industry projects (market-driven or contract) where requirements are actually written and stored are desirable. The direct access to the actual requirements artifacts is not as critical as the knowledge of the structure of the requirements information and methods used in managing them. Experienced requirements project managers, requirements engineering as well as software developers and architects are suitable and very welcome to participate in this study.

Interested in improving your requirements engineering process? Try requirement patterns!

Xavier Franch[1], Cindy Guerlain[2], Cristina Palomares[1], Carme Quer[1], Samuel Renault[2]

[1] GESSI Research Group, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
{franch, cpalomares, cquer}@essi.upc.edu
[2] SSI Department, CRP Henri Tudor, Luxembourg, Luxembourg
{cindy.guerlain, samuel.renault}@tudor.lu

Introduction

Requirement elicitation is the process of acquiring the system requirements from the system stakeholders. This process is critical in all software projects: if not all the requirements are elicited, or if some elicited requirements do not describe real stakeholder needs, or if the quality of the requirements is poor (e.g., they suffer from ambiguities), the chance of project failure increases.

Techniques supporting requirements elicitation (interviews, meetings, story-boards...) are mostly oriented to obtain requirements from scratch and they may hardly take advantage of a fundamental observation:

> When specifying a system, it is quite usual that a significant proportion of requirements is recurrent and belongs to a relatively small number of categories, especially in the case of non-functional requirements.

Our motivation is to consider this observation for improving the effectiveness of the requirement elicitation process. We are using the concept of software requirement pattern [1] (SRP). An SRP basically consists of a template that generates one or more requirements, and some information to identify its need in a particular project, and how it may be tailored to this project.

The main benefits of using SRPs may be summarized as: 1) more effective requirement elicitation (requirements are not built from scratch; a process guides the engineer by giving advices, suggesting information, ...); 2) improved quality and consistency of requirements documents (by using a uniform style); 3) improved requirements management (e.g., clear traceability from requirements to goals).

What have we done?

The GESSI-UPC and SSI-CRPHT departments have built a framework, PABRE, for integrating the concept of SRP in the requirements elicitation [2]. PABRE has been designed and validated from the postmortem analysis of real projects in Luxembourg and the vicinity area. As result of this collaboration, PABRE embraces several assets (see http://www.upc.edu/gessi/PABRE/index.html for details):

- A catalogue of 32 SRPs for non-functional requirements (related to concepts like security, performance, availability, etc).
- Tool Support - two subsystems: one for catalogue management, one for pattern application.
- A method for guiding the elicitation process using patterns.

Wanted from industry: which collaboration scenarios are possible?

We distinguish three different scenarios, depending on the desired interaction among all the parties and the exploitation of the offered assets:

- Free experimentation. For organizations interested on exploring our framework. GESSI-SSI will provide demo versions of the catalogue and the tool, the method and off-line training support. The organization will be allowed to use these assets for an established period of time. At the end, the organization will fill a feedback form (short questionnaire) and present in a 2-hour wrap-up meeting.
- Guided experimentation. For organizations interested on using our framework in a specific project without any modification. GESSI-SSI will provide the catalogue, the tool and the method and a 4-hour training to the project team. The organizations will be allowed to use the assets for an established project, and during it we will provide support under request and agreed terms. At the end of the project there will be an assessment meeting and an assessment report of the experience will be written by the organization. Also the requirement book resulting from the project will be made available to GESSI-SSI.
- Assets customization. For organizations that want to customize our assets for their specific context. The scope both in domain and time will be negotiable. The assets to customize may be:
    - the catalogue, both by: a) including new patterns or modifying the existing ones, b) changing the shape that a pattern may take (e.g., to link requirement patterns to test suites or test strategies),
    - the tool, adding or modifying existing functionalities or import/export capabilities (e.g., to export the requirements to some requirement management tool),
    - the method, by thinking on a specific context of use.

  In all of these situations, an individual study will be made to determine the details of the collaboration (e.g., schedule, charging schema, etc.).

In all of the scenarios, the collaboration will be formalized with a signed agreement.

References

1. J. Withall. *Software Requirement Patterns*. Microsoft Press (2007).
2. S. Renault, O. Mendez, X. Franch, C. Quer: "A Pattern-based Method for building Requirements Documents in Call-for-tender Processes". *International Journal of Computer Science & Applications* (IJCSA), 6(5): 175-202 (2009), available electronically at http://www.tmrfindia.org/ijcsa/v6i57.pdf.

The impact of domain knowledge on the effectiveness of requirements engineering activities

Ali Niknafs, Daniel Berry
David R. Cheriton School of Computer Science
University of Waterloo
{aniknafs,dberry}@uwaterloo.ca

1 Introduction

It is commonly believed that people involved in software engineering practice should possess knowledge of the problem domain at hand in order to be effective. No doubt, domain expertise helps practitioners achieve their goals faster. A domain expert will quickly catch up with development activities, since he/she understands other stakeholders' language and does not waste time learning about the domain.

The perception is that a person lacking domain expertise will slow down the pace of development, as he/she keeps learning about the domain and asking questions while others are busy trying to make progress.

This perception is mostly true, but what happens to creativity? Software engineering is considered a creative activity. While software engineering problems require creative solutions, people with prior experiences in the domain of the problem at hand tend to fixate on the solutions that they have seen before. Hence, knowledge of the problem domain may be a hindrance to the needed creativity.

In addition, each domain expert has tacit assumptions about the domain of the problem at hand that may be inconsistent with other stakeholders' assumptions. These inconsistencies may get coded into the system, and may be discovered late in the lifecycle or never. The later they are discovered, the more expensive they are to resolve and fix.

The question driving this research is “How can we reap the benefits of domain expertise while mitigating its weaknesses and not suppressing creativity?”

Domain ignorance might be a solution in knowledge-intensive activities such as business analysis, requirements elicitation, and inspection, all part of requirements engineering. A domain ignorant may explore a wider range of ideas than a domain expert to come up with a really creative solution. He/she might come up with ideas that a domain expert would never have dreamed of. Also, a domain ignorant is able to state ideas independent of any domain assumptions and ask revealing questions that could lead to exposing issues that domain experts have overlooked or to exposing inconsistencies among other stakeholders' assumptions. In fact, we hypothesize that for knowledge-intensive activities, a team with a mixture of domain expertise and domain ignorance outperforms both a team with only domain experts and a team with only domain ignorants. We want to do experiments to test this hypothesis in various ways.

2 Wanted from Industry

We would like to examine system requirements and other artifacts from past, current, and future projects. With a past project, perhaps we could determine after the fact the distribution of domain knowledge in the project and correlate that distribution with measures of the project's success.

With a current or future project, we would like the opportunity to observe the effects of different distributions of knowledge on the effectiveness of the project's team. With any project, we would use various data collection techniques, including interviews, questionnaires, and observations.

We would like also to be able to attend an occasional requirements engineering activity such as an elicitation or inspection session and to control the distribution of domain aware and domain ignorant people participating in the session and then to measure the effectiveness of the team.

3 Benefit to Industrial Participant

If the hypothesis is supported, then the participating organization will know it before any paper describing the results is published and will be able to use this knowledge to its benefit before others will know it.

References

1. Berry, D. M.: The Importance of Ignorance in Requirements Engineering. Journal of Systems and Software 28 (2), pp. 179-184, (1995)

Towards a Universal Syntax of Software Requirements

James Hulgan and Joy Beatty

Seilevel, Inc.
3410 Far West Blvd., Suite 350
Austin, TX 78731
james.hulgan@seilevel.com, joy.beatty@seilevel.com

Abstract. The variety and complexity of natural languages can often result in missed or poorly implemented requirements. We intend to discover whether there exists a possible syntactic structure, or rules for syntactic structures, which lend themselves to more understandable requirements sentences. Below, we propose analyzing English-language requirements statements used in the industry and determining whether there exists some correlation between statement complexity and the success rate with which the requirements are implemented. In doing so, we hope to determine whether there is an ideal English syntax for capturing software requirements such that they are more easily understood by stakeholders, and consequently have a better chance of successful implementation.

Keywords: Software Requirements, Requirements Engineering, Requirements Syntax, Requirements Language

1 Introduction

A poor understanding of requirements at any phase of a project leads to missed or incorrectly implemented requirements. Consequently, such projects incur additional expense due to rework, business adoption fails, and return on investment is lower. Furthermore, virtually all current software development projects are global in nature, involving distributed teams with various sociolinguistic backgrounds. Business stakeholders, requirements practitioners, test resources, and support personnel often possess varying degrees of ability in the language in which the requirements on any given project are written. Requirements written in a manner such that they are easily understood by speakers of various languages can help avoid expensive misunderstandings. To do so requires balancing the complexity of the requirements statement in order to facilitate better comprehension with the need to fully communicate the requirement. We propose to determine whether there is a way to a structure requirements statement such that it facilitates enough of an understanding for speakers of various languages while preserving enough meaning to ensure that the requirement is correctly implemented. In doing so, the scope of the proposal is focused on the syntax of individual requirements statements written in English to determine which structures facilitate better understanding of the requirement and implementation of requirements. To further this end, both empirical research and leverage of current literature in linguistics is needed in order to determine answers to the following questions:

- Are there valid syntactic structures in English that lend themselves to better comprehension by speakers of various languages?
- What is the best tradeoff between syntactic complexity, understandability, and semantic completeness in order to ensure that requirements are implemented correctly?
- Are certain structures more easily understood by those with computer science/development backgrounds?
- Are there any side benefits to developing certain structures (e.g., can they be used to develop knowledge bases or facilitate requirements reusability?)

1.1 Data That Can be Provided by Practitioner

As industry practitioners, we have access to and will provide the following data in order to assist with the study:

- Examples of requirements written for various types of audiences—including requirements written in English for non-English speaking audiences.
- Requirement authors and developers that can be interviewed or surveyed.
- Supplemental material provided in addition to requirements statements (use cases, models).
- Approximate number of review cycles required for each document.

1.2 Background Expertise Required by Researcher

We are looking for researchers with expertise in, experience with, or a strong desire to learn the following subject matter:

- Basic understanding of linguistics and formal languages. Some work in or a strong desire to learn about psycholinguistics or computational linguistics.
- Familiarity with methodologies for analyzing document complexity, analyzability, and associated metrics.

Intercultural Requirements Engineering for Software Development: Culture and its Impact on Requirements Negotiation

Georg Herzwurm[1], Mareike Schoop[2], Benedikt Krams[1]

[1] Department for Business Administration and Information Systems, esp. Business Software, University of Stuttgart, 70174 Stuttgart, Germany
{herzwurm, krams}@wi.uni-stuttgart.de

[2] Information Systems I, University of Hohenheim, 70593 Stuttgart, Germany
m.schoop@uni-hohenheim.de

Introduction

As far as globalization and internationalization forces companies to close ranks, to build networks and to collaborate, the effect on Requirements Engineering (RE) practices used in Software Development needs to be challenged. Previous work has concentrated on culture and its dimensions ([1], [2]), on behavior and the importance for international collaboration [3] as well as on the effect of culture on negotiations ([4], [5], [6]), but there are no findings adopting this knowledge on negotiations between customers and developers in the field of RE.

The study

Academia can provide coherent solutions for industrial practice to professionalize negotiation processes between customers and developers, to avoid failure and to reach sound and stable win-win situations to reduce requirements changes to the necessary minimum by examining intercultural behavior. In other words (*research question*): *does knowing the cultural background of the negotiation counterpart in globally distributed Software Development lead to better negotiation results* (where "better" needs to be interpreted by the measure "amount of change requests relative to the overall amount of requirements")?

Evaluating culture and behavior is complex because of the "soft" nature of the term culture and gaining profound results needs well-chosen empirical research items. Therefore an all-encompassing approach cannot be presented at this stage of research; asking the right questions is highly dependent on the precise activities of the examined companies or their business units. Nevertheless the study can be divided into the following artifacts:

1. Actual state analysis: how are requirements negotiated?
   Result (R): process description
2. How will the cultural background of negotiation involved parties be measured?
   R: cultural clusters[1]
3. What is the impact of culture on the different negotiation process steps?
   R: mapping outcome with the cultural clusters
4. Target state: how can the negotiation process be improved by cultural knowledge?
   R: e.g. cultural success factors for each process step

Wanted from industry

Research partners from industrial practice should be involved in global distributed Software Development projects with a high rate of cultural diverse participants no matter what size the company has. As far as organizational culture affects members' behavior of this organization, companies working together with legal independent entities are welcome (e.g. established by sourcing). In concrete terms potential partners need to:

- Develop Software
- Support a high rate of customer-involvement during RE processes
- Have different cultures in their teams involved
- Want to professionalize their requirements negotiation processes
- Want to minimize change requests at a late stage in the RE process

References

1. Hofstede, G.: Culture's Consequences, Comparing Values, Behaviours, Institutions, and Organizations Across Nations, 2nd ed., London (2003)
2. Hofstede, G., Hofstede, J., Minkov, M.: Cultures and Organizations: Software of the Mind, 3rd ed., New York (2010)
3. Imai, L., Gelfand, M.J.: Interdisciplinary perspectives on culture, conflict, and negotiation, in: Bhagat, R.S., Steers, R.M. (eds.): Cambridge Handbook of Culture, Organizations, and Work, Cambridge (2009)
4. Grünbacher, P., Seyff, N.: Requirements Negotiation, in: Aurum, A., Wohlin, C. (eds.): Engineering and managing software requirements, Heidelberg (2005)
5. Koeszegi, S.T., Vetschera, R., Kersten, G.E.: National Cultural Differences in the Use and Perception of Internet-based NSS - Does high or low context matter?, in: International Negotiation, Vol. 9 (1), pp. 79 – 109, (2004)
6. Kersten, G. E., Koeszegi, S.T., Vetschera, R.: The Effects of Culture in Computer-mediated Negotiations, in: JITTA - Journal of Information Technology Theory and Application, Vol. 5 (2), pp. 1 – 28, (2003)

[1] e.g. by cultural society, nationality, mother tongue, domicile, training place, academic education inter alia as well as combinations of these domains

Higher Efficiency through Tailored Requirements Processes in Reuse-oriented Development

Sebastian Adam

Fraunhofer IESE, Fraunhofer Platz 1, 67663 Kaiserslautern
{sebastian.adaml}@iese.fraunhofer.de

1 Introduction

Even though reuse in general and product lines (PL) in particular have been proven to be a promising strategy to fasten software development, developing new software applications based on existing artifacts is often less efficient than expected.

Among others, one important reason is the non-systematic mapping of customer requirements to reuse capabilities. On the one hand, current reuse-oriented RE approaches rather foster the direct reuse of anticipated requirements than the effective alignment of actual needs with available components. Thus, especially for systems where a high degree of customizability is required, these approaches lead to significant iterations, because a large number of individual requirements has to be addressed in order to allow the customer to stand out from the competition. On the other hand, eliciting customer requirements from scratch without considering any reuse capability early on is also not an appropriate option. Particularly since reuse implies a certain set of constraints, it becomes apparent that not all customer requirements can be realized as initially stated. Rather, trade-offs between ideal requirements and rapid development benefits must be made, leading also to costly rework iterations. However, making this trade-off is challenging, because information about the realizability of requirements is often neither formalized nor available in the early requirements phase. Requirements elicitation therefore becomes an error-prone task, and it relies on experts to predict the impact of non-anticipated requirements. Thus, checking the fit in an analytical manner and improving the requirements afterwards is still the state-of-the-art, and approaches, which better cope with this challenge, have not been provided yet. Hence, it is still hard to elicit new requirements while considering reuse characteristics early on.

In our previous research, we have introduced the notion to tailor requirements processes based on the characteristics of a given reuse infrastructure. These tailored processes should enable requirements engineers to use externalized knowledge about the infrastructure for guiding the elicitation and negotiation in a better informed way.

2 Wanted from Industry

While we have already shown the general feasibility of this idea, we are now looking for a case study organization (ideally from Europe and the domain of information systems) that develops software in a reuse-based manner by either combining components from a repository, or by configuring a core product platform. After making a base lining by performing some interviews, we plan to conduct a RE process tailoring, in which platform architects, developers, and development process owners should participate. The result of this tailoring will be a customized RE process guideline that is expected to lead to **significant improvements in the organization especially with regard to requirements coverage and required rework**. For checking this hypothesis according to a measurement plan that we will provide, the process guideline should therefore be used by the case study organization in at least one customer development project after the tailoring workshops.

Identifying Value-Based Criteria for Requirements Triage and Selection Decision-Making

Mahvish Khurum, Tony Gorschek

Blekinge Institute of Technology, Karlskrona, Sweden

mkm@bth.se, tgo@bth.se

1 Introduction

In software intensive products such as cars, industrial robots or telecom systems, software has traditionally been associated with cost, and there has been no real perception of its value in relation to the entire product offering. However, as software is becoming a larger part of the main competitive advantage, driving innovation and product differentiation while hardware is becoming more standardized, the valuation of software is becoming critical. A consequence of this is that software intensive products in general, but the software parts in particular, are increasing in size and complexity, and the cost associated with the development of the software part is also increasing. Decisions taken regarding the software will impact the entire product's life cycle and value. Thus, it is important to take into consideration value aspects impacting both short-term and long-term success of the product and company.

1.1 Objectives

The objective of the study is to identify which value aspects are considered when taking requirements triage and selection decisions. To fulfill this objective, value aspects considered currently need to be elicited from the industry participants and value aspects that should be considered (ideally) need to be elicited. In addition we want to elicit how the value aspects are and should be evaluated/assessed/measured.

RQ1: Which value aspects are considered when taking requirements triage and selection decision?

RQ2: How the value aspects are evaluated/assessed/measured?

RQ3: Which value aspects should be considered when taking requirements triage and selection decision?

RQ4: How the value aspects should be evaluated/assessed/measured?

3 Method

We propose to use exploratory case study method. We propose to interview all the roles involved in the decision-making (minimum two persons per role) per product. The participants would first be asked to name and briefly describe value aspects they consider today. Information about how these value aspects are evaluated/assessed/measured would also be elicited. In the second step, participants will be presented with an exhaustive list of value aspects to elicit value aspects that should be considered and how these should be evaluated/assessed/measured.

3.1 Wanted from Industry

- ***Context***: Market-driven software product development companies, medium to large-scale companies
- ***No. of products***: 2-4
- ***Roles***: People involved (directly or indirectly) in requirements triage and selection decision-making. For example, product managers, requirements managers, system managers, marketing units
- ***No. of personnel***: 2-4 people in one role/per product as it is essential for triangulation
- ***Artifacts***: access to artifacts used in/for requirements triage and selection decision making for example, requirements specification documentation, roadmaps, product strategies (if used for requirements triage and selection)

7 Empirical Live Experiment

Editors

Jörg Dörr
Fraunhofer IESE, Germany, joerg.doerr@iese.fraunhofer.de

Technical Programme

REFSQ 2011 Live Experiment about Risk-Based Requirements Prioritization: The Influence of Wording and Metrics

Andrea Herrmann

University Heidelberg, Germany
AndreaHerrmann3@gmx.de

Abstract **[Context & motivation]** Risk is one possible criterion for prioritizing requirements. **[Question/problem]** Although risk is often recommended as a prioritization criterion, little is known about the practical challenges of the estimation of IT-related risk. At REFSQ 2011, a live experiment was executed. It investigated the influence of the following two factors on the quality of risk estimation: The fuzziness of the wording of Misuse Cases and requirements, and the metrics used for quantifying probability of the Misuse Case to happen and damage caused. This experiment is number 8 in a series of experiments about the estimation of IT-related risk. **[Results]** The two factors investigated had an effect on the quality of the risk estimates: Quantified wording leads to better risk estimates. Relative probability and damage on point scales are easier to estimate than absolute values in €, hours or %. **[Contribution]** The effects observed are not surprising, but this is the first experiment investigating these effects empirically and so it is the first time that the effect is scientifically observed and quantified.

Keywords: experiment, requirements prioritization, risk estimation

1 Introduction

Several criteria can be used to quantify a requirement´s priority, like the implementation cost caused by this requirement, business value added, or risk. Many requirements are demanded in order to reduce risk. This is especially the case when requirements have been elicited using Misuse Cases (MUCs), because such requirements engineering methods identify risk-reduction requirements, which are usually called countermeasures.

Although risk is often recommended as a prioritization criterion, little is known about the practical challenges of the estimation of IT-related risk. At REFSQ 2011, a live experiment was executed. This experiment is number 8 in a series of experiments about the estimation of IT-related risk. The first two experiments have been published in the Journal of Empirical Software Engineering in 2009 [12]. These experiments raised several research questions concerning the task of risk-based requirements prioritization. Meanwhile, further experiments have been executed to answer some of these questions. The objective of the experiment series is to empirically identify the

factors which influence the quality of risk estimates and to understand the practical challenges and needs of risk estimation.
The present experiment investigates the influence of two factors on the quality of risk estimates and whether this quality can be predicted by the estimators. The research questions are:

1. **Fuzziness:** Are the estimates achieved for quantified Misuse Cases and requirements better than those for fuzzy or abstract ones?
2. **Metrics:** Are damage and probability of risks easier to estimate on a scale of points than in percent or Euro?
3. **Predictability of uncertainty:** Can estimators judge or even quantify the certainty of their estimates?

The paper is structured as follows: Section 2 motivates risk-based requirements prioritization and the research questions, based on related work, Section 3 describes the experiment design and Section 4 its execution. Section 5 summarizes the results and interprets them. Section 6 is dedicated to the validity discussion, and Section 7 summarizes the results.

2 Motivation and Related Work

MUCs have shown to be a good means for eliciting requirements, which then have the function of a countermeasure [11], [21], [22]. That means that they mitigate or prevent a MUC from happening (i.e. reduces the MUC´s probability) or at least detect its execution and so can reduce the damage caused. Consequently, it seems to make sense to quantify the countermeasure´s priority by estimating the risk reduction which it achieves. When such a requirement is implemented, the risk of a MUC must be lower than without the requirement being implemented. This risk reduction is the benefit which the requirement adds to the system. Risk commonly is quantified by the product of probability of a MUC (or risk event) with the damage it causes (e.g. in [13]).

Risk estimation has been used for countermeasure prioritization in security engineering (e.g. by [1], [25]). In requirements engineering, risk has been proposed as a criterion to prioritize requirements for instance in WinWin [16], in the DDP method of the NASA [5], [6], [7], [8]) and by [2] as well as [15]. However, there was no critical empirical evaluation of the risk estimation task, neither in the security domain nor in requirements engineering. So, there is no documented experience on how well people can estimate risk and what practical challenges are encountered in risk estimation.

In my own work, I have applied MUCs for eliciting requirements related to all types of non-functional requirements, not only to security [11]. As this worked well, the next step was to prioritize non-functional requirements based on risk reduction. In 2007, we have executed two initial pilot experiments about risk-based requirements prioritization [12], which raised a list of fundamental research questions about risk estimation. Meanwhile, a series of six further experiments about the risk estimation of IT-related risks have been executed in order to investigate these research questions. Most of these experiments have not yet been published (but their publication is in preparation). All experiments have shown that risk estimation is difficult and the experiments revealed what exactly makes risk estimation so difficult and what helps estimators to do better estimations. Such factors have been:

- Discussion in the group takes more time, but the estimators then feel more certain about their estimates.
- Transparency (i.e. to see how the probability and damage estimates influence the resulting priorities) improves the ease of use and the participants´ feeling to produce realistic results.
- Clear prioritization criteria, tangible and quantified MUCs and requirements lead to a lower coefficient of variation of the estimates among participants.
- Learning by feedback improves estimates, but age or experience of the estimator play a minor role.

However, risk-based requirements prioritization has also revealed strong disadvantages compared to simpler prioritization methods:

- It takes more time because more decisions must be made (at least four values must be estimated for prioritizing one requirement: two probabilities and two damage values) and each of these estimations takes more time than prioritization decisions made in simpler methods.
- Risk estimation is experienced as less easy to use; participants feel less certain about their results.
- Estimated probabilities deviate by several hundred percent from the real value.
- Former experience with a risk event leads to an even larger overestimation of probability.
- Resulting priorities are not better than those obtained by other methods: They reflect the view of participants less, show a higher standard deviation and the repeatability of estimations is not better.
- Risk estimation demands a lot of information (as published in detail in [10], based on several former experiments).

While there is no empirical quantitative evaluation of risk estimation in the field of IT requirements prioritization, there is much (risk) estimation experience in other scientific fields, like health research, military, and psychology. All authors agree that risk estimation is inherently difficult for humans, especially estimating probabilities. Human estimations are always biased. Many biases are known [18] which lead to bad estimations of probabilities, frequencies and values, or as Tversky and Kahneman [23] put it: "intuitive predictions and judgement under uncertainty do not follow the laws of probability or the principle of statistics. Instead, people appear to rely on a limited number of heuristics and evaluate the likelihood of an uncertain event by the degree to which it is representative of the data generating process, or by the degree to which its instances or causes come readily to mind."

There are few empirical publications related to the research questions we investigate in the present experiment. An experiment has shown that relative values (of requirement importance) can more easily be estimated than absolute (cardinal) ones [14]. Gigerenzer [9] in several studies investigated how people - especially doctors and lawyers - cope with probabilistic thinking. He found that the human brain better copes with integer numbers (e.g. “7 out of 1000 patients”) than probabilities in %. One observes also subtle differences between frequency and probability: A qualitative study evaluates the response of women to various formats used in the communication of breast cancer risk. Graphic discrete frequency formats using highlighted human figures had greater salience than continuous probability formats using bar graphs [20]. However, even quantifying how often something has happened within a certain period of time – even if it was not long ago – is difficult [3].

Not only from scientific experiments, but also from practitioners, it is signalled that absolute values of probabilities and damages cannot be estimated well. This is given as the reason why probabilities and damages are estimated in discrete categories, in order to not pretend a precision of the data which does not exist. Two reasons why estimation of probabilities is problematic in practice are: A sufficiently reliable basis is often missing and the interpretation of probabilities is often unclear [4].

These sparse hints plus our previous experience with risk estimation experiments lets us expect that the answers to all of our three research questions could be “yes”.

3. The Experiment Design

The **participants** of the experiment were the conference participants of REFSQ 2011. The participants had to estimate probabilities and damages for several MUCs – in a scenario where a specific countermeasure is not implemented and then when it is implemented. MUCs and requirements were chosen that are familiar to all participants. This is a difficult constraint when the participants come from all over the world and from different work contexts. Therefore, two **types of MUCs** were chosen: (1) Laptop crashes (because it can be assumed that world-wide, similar laptops are used and there is no major national difference in how often laptops crashes), and (2) MUCs when using Google (www.google.com). We assumed that knowledge workers around the world use Google regularly. The probability of a laptop crash is known: It is 7.2% within the first year and 31% within the first three years [19]. However, values for different brands can differ from this average.

It was decided to put the participants in a similar **context of use** by defining the following conditions:

- The estimators adopt the perspective of the user.
- Imagine that your laptop is the only computer which you use for your work. You have many important deadlines, at least one per week. For delivering your work results, you need to do an online research, e.g. using Google.
- You are an experienced user – both of your laptop and of search engines. You are highly motivated to do your work successfully and in an efficient way.

In order to answer the three research questions, we had to use fuzzy and abstract MUCs and requirements, and such which are quantified. In order to compare both, two different questionnaires were prepared, and the participants would be separated in two groups. Group 1 had to cope with the fuzzy wording and Group 2 got the quantified ones. The results of both groups were evaluated separately and compared to each other.

Different metrics were used. Among those found in literature, we chose the following: Probability was estimated in %, times per year and on a scale of 1-7 points, damage was estimated in €, hours of time lost, on a scale of 1-7 points and the dissatisfaction in 1-7 points. The metrics were defined as follows:

- Probability of MUC in % of the number of uses: In 100 work sessions, how many times does the MUC happen? E.g. 10% = 10 out of 100 times.
- Average times per year: Within the period of one year, how often does the MUC happen? E.g. once per year (=1) or once out of 10 years (=0.1).
- Probability in 1-7 points: 1 = extremely unlikely, (almost) never; 3 = rather rarely; 5 = rather frequently; 7 = extremely frequently, (almost) always.
- Damage to you in €: How much money in € (or $, then please indicate) do you as a user of the laptop/ Google lose when the MUC happens?
- Damage to you in hours of time lost: When the MUC happens, how much time do you lose (calendar time, not work time)? That means: When you must wait 24 hours and repeat an activity of half an hour, then you have lost 24 hours of calendar time, but only half an hour work time.
- Your dissatisfaction in 1-7 points: 1 = extremely satisfied; 3 = rather satisfied; 5 = rather dissatisfied; 7 = extremely dissatisfied.
- Damage to you in 1-7 points: 1 = (almost) no damage; 3 = rather low damage; 5 = rather high damage; 7 = extreme damage.

In order to investigate the third research question, the participants were asked to quantify the uncertainty of their estimates in %, points, respectively for each metric. They were also asked how certain they feel about their estimations for each metric.

The research questions use the term "quality of estimates". Like in former experiments, we here used several criteria for quantifying this estimation quality:

- The participants indicate that the estimation was easy to do.
- The participants indicate that they feel certain about their estimates.
- The participants expect that their estimates are reasonable, realistic and useful:
- The data show a low coefficient of variation among participants – if this coefficient is low, the participants agree better.

The material distributed to the participants had the following parts:

a) Questions about the profile of the participant, like age and working experience,

b) Risk estimation tables where the participants estimate probabilities and damages of MUCs – without and with the requirement implemented (the ten MUCs and nine requirements are listed in Table 3),

c) Questions where participants give their opinions on the ease of use of the method and the certainty with respect to the metrics used,
d) Handout of screenshots and explanations/ definitions of the MUCs and requirements used in the experiment.

As the participants were separated in two groups – Group 1 with the fuzzy wording, Group 2 with quantified MUCs and requirements – the questionnaires partly were different. These were parts (b), and (d) where MUCs and requirements were only partly the same, but most often, different versions of the same MUC/ requirement were used.

The questionnaires were pretested with respect to time need and understandability. First, the author tested the questionnaires herself, and then, two persons with software engineering background (but who are not requirements engineering experts) pre-tested the questionnaire.

4. Execution

This experiment took place on 28 March 2011 at the REFSQ 2011 conference. The agenda of the experiment session was the following:

- 14:00-14:20 introductory presentation about risk-based prioritization in general and the experiment especially (the MUCs and requirements to treat, the context of use assumed, definition of metrics used)
- 14:20-15:00 execution of the experiment: risk estimation on paper questionnaires
- 15:00-15:30 presentation of results of former experiments and discussion

The results of former experiments were presented after the experiment, because otherwise they might have influenced the participants´ experiment execution.

The estimators got 40 minutes time. Four participants needed 5 more minutes.

During the experiment, many questions were asked. The most important of these were:

- When in a MUC there is no time period given, does this mean “in a lifetime”? (Answer: yes)
- Do I take the perspective of a fictitious user or my own? (your own, except for the assumptions about the context of use given in the handout)
- What financial damage causes the loss of one hour? (take your own hourly prize)

In the discussion session after the estimation, the following remarks were made:

- There was some uncertainty about what to do.
- The time units used (for referring probability in % to) should be different for each MUC/ risk. (Yes, but this would make comparison and prioritization more difficult)
- Does in MUC2 “3 years” include the first year? (yes)
- There are different assumptions which one could make.

- It was difficult to say whether some probabilities increase or decrease with the laptop´s age: There is deterioration of the laptop/ software, but also factors which improve it.
- One should use empirical data about cost, duration and frequencies, instead of estimating.
- One should use conditional probabilities.

5. Results

After the experiment, 50 questionnaires were gathered – 28 for Group 1 and 22 for Group 2. Five questionnaires were incomplete (four of them belong to Group 1 and one to Group 2).

This Section 5 summarizes the results of the experiment. Section 5.1 gives the profile of the participants, Section 5.2 summarizes the benefits and priorities determined from the estimated probabilities and damages. The subsequent subsections answer the three research questions.

5.1. Participant profile

The participants´ profiles are given in Table 1 and Table 2. As can be seen from these tables, the estimators are quite experienced in the computing domain, in using laptops as well as Google, and many have already experienced the MUCs used in the experiment.

Table 1. Participant profile

Attribute	**Age** (in years)	**Working experience in computing** (in years)	**How often do you use your laptop?** (times per year)	**How often do you use Google?** (times per year)
Minimum value	24	2	40	200
Maximum value	68	49	3000	15,000
Median of Group 1	31	15	360	365
Median of Group 2	31	11.5	350	365
Average on Group 1	35.7	15.3	411	1,438
Average on Group 2	37.0	15.1	435	1,864
Total average	36.3	15.2	422	1,633

Table 2: Previous experience with the MUCs: How many participants (in %) have had experienced each of the MUCs?

Misuse Case	MUC1	2	3	4	5	6	7	8	9	10	number/ participant
Group 1	29%	50%	71%	32%	32%	61%	54%	64%	46%	68%	5.1
Group 2	18%	55%	55%	55%	50%	91%	46%	46%	46%	73%	5.3
Total	24%	52%	64%	42%	40%	74%	50%	56%	46%	70%	5.2

Tables 1 and 2 show that both groups are practically equal with respect to age, working experience, frequency use of laptop and Google as well as their experience with the MUCs.

5.2. Estimated benefits and priorities

This section summarizes and analyzes the estimated benefits and priorities. They do not directly answer our research questions but give the reader an impression of what the results look like and about their quality. Due to lack of space, we do not list the individual estimates of the participants and not the averages of the estimates of all participants, but only the benefits and priorities derived from these.

The benefit of each requirement is the difference between the MUC risks in two situations: (1) When the requirement is not implemented and (2) when it is implemented In situation (1), the risk must be higher than in situation (2). The difference between these two risks equals the benefit of the requirement with respect to a MUC. From the metrics used in the experiment, we chose the following to calculate the MUC risks, requirement benefits and priorities:

- Probability in points on a scale of 1-7 points,
- Damage as the dissatisfaction on a scale of 1-7 points.

These two metrics were chosen because the participants trusted most in the estimates on these two metrics (as shown in Section 5.4) than the others. The risk then can take values between 1 point and 49 points, and the benefit of a requirement can be between 0 and 48. It should not be negative, however, as the requirements were chosen in a way that they should reduce either the probability of the MUC to happen or the damage caused. Therefore, we use the number of negative benefits, which the participants obtain, as a measure for the inconsistency of their results (see Section 6). Table 3 summarizes the benefits and priorities. The priorities were determined comparing the benefits within each group. The requirement with the highest benefit received the highest priority, i.e. the rank 1.

Table 3: Benefits and priorities resulting from the probability and damage estimates. Remark: The two groups got different questionnaires and did not estimate the same MUCs and requirements.

MUC	Requirement	Benefit, Group 1	Benefit, Group 2	Priority, Group 1	Priority, Group 2
MUC1: Your new laptop crashes within the first year	R1: daily data backup		1.20		
MUC2: Your new laptop crashes within the first three years	R1: daily data backup		3.20		
MUC3: Your laptop crashes this year.	R1: daily data backup		5.63		2
MUC1: Your new laptop crashes within the first year	R2: regular data backup	1.93			
MUC2: Your new laptop crashes within the first three years	R2: regular data backup	4.49			
MUC3: Your laptop crashes.	R2: regular data backup	5.04		4	
MUC4: Under load, the performance of the Google search becomes low.	R3: more efficient load balancing	6.24		3	
MUC5: Under load, the Google search takes 3 seconds instead of 0.2 seconds.	R5: double performance by load balancing		2.38		6
MUC6: A typing error in the search string leads to an (almost) empty list of hits.	R6: Automated detection and correction of typing errors.	6.99	3.16	2	4
MUC7: You have found a list of hits and opened 20 of these in a new window/ tab. Then, your browser crashes and all these pages get lost. You must restart your search.	R4: Technical possibility to mark interesting hits and to save the list on your computer.	4.61	3.14	5	5
MUC8: You want to search on web pages are in another language than the search string.	R7: You can define the language of the web pages searched.	2.85		6	
MUC9: You work search for an English search string, but you want to find web pages in another language	R7: You can define the language of the web pages searched.		3.22		3
MUC10: You search for pages	R8: Combination of	7.06		1	

which contain all of three key words, and additionally one out of three synonyms. However, Google also finds pages which contain only one or two of the key words.	key words is possible.				
MUC10	R9: Google "advanced search" allows the *and* and *or* combination of key words.		7.80		1

Do the two groups agree with each other about benefits? As we see in Table 3, the two groups agree in many cases:

- MUC10 has a high probability and the countermeasure proposed is quite effective, therefore R8 respectively R9 has the highest benefit and gets the highest priority.
- R6 achieves a similar benefit in both groups.
- R7 has about the same benefit with respect to MUC8 and MUC9 (which are quite similar). The resulting priorities, however, are differ by 3 ranks because the priorities depend on the benefit which were attributed to the other requirements on the list.
- The benefit of a backup (R1 or R2 respectively) is quite low when the laptop is new, but grows by time. R1 has a slightly higher benefit than R2. This makes sense when R1 is probably executed more frequently than R2.

The only disagreement between the groups concerns the benefit of R6 against MUC6: Group 1 attributes R6 a benefit which is double the size of that which Group 2 gives it, although in this case, the wording is exactly the same for both groups.

Table 4: Probability estimates for MUCs 1 and 2 (assuming that no backup is implemented).

Misuse Case	**Group**	**Probability of MUC in % of the number of uses**	**Average times per year**	**Probability in 1-7 points**
MUC1: Your new laptop crashes within the first year	Group 1	1.90	1.78	2.00
MUC1: Your new laptop crashes within the first year	Group 2	3.89	0.89	1.41
MUC2: Your new laptop crashes within the first three years	Group 1	6.31	3.06	3.14
MUC2: Your new laptop crashes within the first three years	Group 2	11.72	1.37	2.62

In order to judge the estimates´ quality, it is also interesting to compare the estimates with statistics if available. The probabilities of a laptop crash are known: It is 7.2% within the first year and 31% within the first three years [19]. We want to compare these probabilities with the estimates from the experiment, as given in Table 4. Therefore, we need to transform the probability per year into a probability per 100 uses by dividing it by the number of uses. If the laptop is used 422 times per year (what is the average from Table 1), this means that MUC1 happens in 1.7 and MUC2 in 2.4 out of 100 uses. We see in Table 4: The estimates in % of uses for MUC1 approximately correspond to the statistical value – especially for Group 1–, but the probability of MUC2 is overestimated.

So, we see that the estimated values are in a realistic order of magnitude and the two groups agree mostly with respect to benefit estimates and priorities of the requirements.

5.3. Fuzziness: Are the estimates achieved for quantified Misuse Cases and requirements better than those for fuzzy or abstract ones?

For answering this question, we compare the results of Group 1 and Group 2 with each other. Group 1 had the rather fuzzy MUCs and requirements, while Group 2 got the quantified ones. We expect that the estimates of Group 2 should be better than those of Group 1, and wonder by how much. As was said in Section 3, we use four criteria for measuring the quality of the estimates. Table 5 presents the results with respect to the first three of these criteria. The responses of the participants were coded as follows:

- The participants indicate that they feel **certain** about their estimates: very certain (+2), rather certain (+1), undecided (0), rather uncertain (-1), very uncertain (-2)
- The participants indicate that the estimation was **easy** to do: very easy (+2), rather easy (+1), undecided (0), rather difficult (-1), very difficult (-2)
- The participants expect that their estimates are **reasonable, realistic and useful**: very reasonable, etc. (+2), rather reasonable, etc. (+1), undecided (0), rather not reasonable, etc. (-1), not reasonable etc. at all (-2)

For testing the statistical significance of the differences observed between Group 1 and Group 2, we used Pearson´s chi-square test of independence [17]. This test found that there is no statistical significance between the judgments of the two groups, not within a significance level of 90%. Only for the certainty about estimating probability in 1-7 points, the two groups´ judgment differs with a certainty level of 90%. The differences are not statistically significant due to strong variations of the judgments within the groups. Only the fact that all effects show the trend that Group 2 felt better about their estimates with respect to each criterion, is a weak evidence to support that the answer to our research question number 1 is "yes".

These judgments are comparable with those found in previous student experiments. Compared to the experiments in [12], this experiment is most similar to Method 2 in Experiment 2 there, where we found as average judgments: ease of risk estimation = -0.92 points (comparable to Group 1 here), results are reasonable, realistic and useful = 0.17 points (comparable to Group 2 here).

Table 5: Answers to the questions "How certain do you feel about your estimations of probability/ damage?", "Is risk estimation easy to execute?", "Are probabilities/ damages easy to estimate?" and "Do you expect that your estimates are reasonable, realistic and useful?".

	Certainty of probability estimates			**Certainty of damage estimates**				**Ease of estimation**			**Reaso-nable, realistic, useful?**
	%	Times/ year	1-7 point	€	Hours lost	Dissatis-faction 1-7 pt	1-7 pt	Risk	Probability	Damage	
Group 1	-0.16	-0.2	0.24	-0.79	0.04	1.08	0.38	-0.92	-0.72	-0.76	-0.24
Group 2	0.23	0.32	0.95	-0.57	0.09	1.32	0.77	-0.55	-0.41	-0.50	0.23

The fourth quality criterion for the estimates is the coefficient of variation among participants – if this coefficient is low, the participants agree better. The coefficient of variation is defined as the ratio between standard deviation and average of a sample. Dividing the standard deviation by the average has a normalizing effect. For Group 1, averaged over all participants, MUCs and metrics, the coefficient of variation is 1.652, while for Group 2 it is 1.319. So, for Group1 it is larger than for Group 2, what means that the estimates of Group 2 are more consistent (and therefore better) than those of Group 1.

Table 6: Coefficients of variation of the estimates – for each group and metric.

Metric	**Coefficient of variation in Group 1**	**Coefficient of variation in Group 2**
Probability of MUC in % of the number of uses	1.655	1.570
Average times per year	2.013	2.361
Probability in 1-7 points	0.588	0.664
Damage to you in € (when MUC happens)	3.131	1.807
Damage to you in hours of time lost	3.097	1.865
Damage to you in 1-7 points	0.653	0.511
Your dissatisfaction in 1-7 points	0.427	0.456

The coefficients of variation of the probability and damage estimates are almost equal for both groups. It is clearly larger for Group 1 only for two metrics: damage in € and damage in hours lost. However, for these two metrics, the coefficient of variation is naturally larger than for the other metrics, because these scales are not limited, while the other metrics allow values only within a given interval. Therefore, this difference not necessarily has a significance.

In order to get a feeling of what a coefficient of variation of 0.5 means, imagine: If we have a scale points from 1 to 7, then the average is about 4 points. A coefficient of variation of 0.5 then means that the standard deviation is 2 points, on a scale of 1-7 points. This standard deviation spans quite a large interval, i.e. the estimates vary (in average) from 2 to 6 points. In a former experiment [12], we found for probability and damage estimates (both estimated in %) a similar coefficient of variation of around 0.4.

Although the fourth criterion – the coefficient of variation – does not show a clear difference between the two groups, the first three criteria strongly indicate that Group 1 with the fuzzier wording felt less certain about their estimates, found the estimations less easy to do and expected less reasonable, realistic and useful results. We conclude that the fuzziness of wording has an effect on the estimates.

5.4. Metrics: Are damage and probability of risks easier to estimate on a scale of points than in percent or Euro?

While in the preceding section, we compared the two groups with each other, here we compare whether (within each group) some metrics were easier to use than others. This comparison is also based on the data in Table 5.

Probability estimation: We found no statistically significant difference for the certainty between the probability estimations in times per year or in %. However, in points the probability was easier to estimate than using one the other two metrics. This result was statistically significant at a significance level of 85% for Group 2, but not statistically significant for Group 1.

Damage estimation: For the damage estimation, both groups felt most certain for their estimation in dissatisfaction (measured in points). This metric got more than 1 point in average, what corresponds to “rather certain”. Second was the metric damage in points, then damage in hours of time lost, and most uncertain the participants believed their estimates in € to be. The latter metric got negative votes in average, while “damage in points” and “damage in hours of time lost” got values between 0 and 1. The differences between the subjective certainties of these metrics were found to be statistically significant at a significance level of at least 97.5%, except for these: The difference between dissatisfaction in points and damage in points was not statistically significant and also the difference between hours lost and €. As we see in Table 6, the coefficient of variation is much larger for the estimates on € and in time lost than for the other metrics.

We conclude that probability is easier to estimate in points than in % or times per year. Damage is easier to estimate in points (either measuring dissatisfaction or damage) than in hours lost or €. Consequently, we recommend using point scales for estimating both probability and damage of risks.

5.5. Predictability of uncertainty: Can estimators judge or even quantify the certainty of their estimates?

For answering this question, we compare the actual uncertainty of the estimates with the uncertainty expected by the estimators. The actual uncertainty is measured by the standard deviation of the estimates of damage or probability (for each group separately). The expected uncertainty is given by the answer to two questionnaire questions:

1. "How certain do you feel about your estimations of probability/ damage?" (on a scale of -2 to +2 points)
2. "Please, estimate by how much the probability/ damage might be uncertain." (for each metric, in times, €, points, etc)

The first criterion shows whether the participants can judge which of their estimates are better (in terms of metrics used), while the second criterion shows whether the participants can even quantify their estimates´ uncertainties.

In Section 5.4, we found that concerning probability estimation, the participants felt most certain about their estimates in points. So, the research question here reduces to: "Is the coefficient of variation of the probability in points lower than of the other two scales on which probability was estimated?" As Table 6 shows, this is the case.

Damage estimation: Participants felt most certain about their damage estimates measured in dissatisfaction (measured in points), second was damage in points, then damage in hours of time lost, and finally the € scale. Is a corresponding order of the size of the coefficient of variation visible? Yes, this is the case (see Table 6), for one exception: For Group 2, damage in € has a slightly lower coefficient of variation than for scale "damage in hours of time lost".

Apart from this latter exception, the estimators knew to judge very well which of their estimates are better than the others.

Table 7: Comparison of the estimates´ standard deviation with the participants´ expected uncertainty. The units are the % points, times per year, etc, corresponding to each metric.

Metric	Standard deviation in Group 1	Standard deviation in Group 2	Expected uncertainty of Group 1	Expected uncertainty of Group 2
Probability of MUC in % of the number of uses	7.288	6.384	23.9	15.5
Average times per year	27.801	62.414	18.7	14.2
Probability in 1-7 points	0.602	0.609	2.6	1.5
Damage to you in € (when MUC happens)	17338.0	4722.2	1785	11793
Damage to you in hours of time lost	555.930	467.815	21.3	33.0
Damage to you in 1-7 points	0.575	0.486	1.5	1.6
Your dissatisfaction in 1-7 points	0.591	0.585	1.9	1.9

Table 7 compares the standard deviations of the estimates with what the participants estimated to be the uncertainty for each metric. Here, we use the standard deviation instead of the coefficient of variation, because we need a measure which is comparable to the uncertainty interval which the participants estimated

The uncertainty intervals which the participants expected do not reflect the actual standard deviations. In some cases when they expected a higher uncertainty, the standard deviation in fact was higher, in many other cases this is not true.

(Remark concerning the expected uncertainty: Here, Group 1 was more optimistic about their probability estimations than Group 2, and for the damage estimations it was vice versa. This difference between the two groups is not statistically significant, however, as the standard deviations of these values are very high and not all participants gave a quantitative estimate of their uncertainty.)

Here, as in Table 5, the probability estimation was expected to be more certain for the point scale and less certain for the two others. For the damage estimate, dissatisfaction in points and damage in points are equally certain, while hours of time lost is much more uncertain and cost in € is most uncertain. This order corresponds to the one found before, except for that the difference between dissatisfaction in points and damage in points vanishes.

So, we find that the participants can judge which of their estimates are more uncertain, but they can not quantify this uncertainty interval.

6. Threats to Validity

The validity of an experiment means that the experiment measures exactly what should be measured. The four aspects of validity are: conclusion validity, internal validity, construct validity, and external validity. In what follows, we distinguish between the validity of the probability and damage estimates, and the validity of the effects which different treatments caused. These different treatments were fuzzy wording versus quantified wording, and the use of different scales for estimating probability respectively damage. While the absolute values of the estimates are not too valid because risk estimation is very difficult, we believe that the effects caused by different treatments are valid.

Validity of the estimates: We have seen above that the probability of MUCs 1 and 2, for which statistical values are known, have – in average - been estimated approximately correctly by the participants. However, estimates vary strongly, for instance plus/minus 2 points on a scale of 1-7 points. This uncertainty is partly due to the fact that in an experiment one always must use fictitious cases, while in practice we expect better results when the estimators are experts with respect to the risks to estimate and when they have statistics available about these risks. Nevertheless, in the result, the participants agree on which MUCs are more important than others and which requirements consequently are important.

As in previous experiments, it happened that the resulting benefit of a requirement was negative, what should not be. This happened when the participants had no tool (e.g. a spreadsheet table) which calculates the benefit from the estimates during the estimation workshop. Unfortunately, due to practical constraints, in many experiments, one cannot provide everyone with such a tool (especially, the supporting computer), so this experiment again worked with paper questionnaires. Due to the limited space which one sheet of paper offers, the estimation for the scenario when the requirement is not implemented and when it is implemented, had to take place on subsequent sheets, it did not fit on the same sheet. This is one possible explanation why estimations can happen to be inconsistent. We measure the validity of the estimates made by the number of times where the resulting benefit is zero or negative. Such numbers show that the participants had cognitive difficulties when filling the questionnaire, which threatens the validity of their risk estimates (but probably not their judgment of how difficult the estimations were to do). When the benefit is zero, this is a hint that the participant did not see the difference between the two questionnaire tables or the two scenarios with and without the requirement implemented. When the benefit is negative, the participant has not understood the principle that with the requirement implemented, either the probability or damage should be lower. Group 1 produced 209 benefit estimates. Among these were 72 zero (=34%) and 13 (=6%) negative. There were two questionnaires, where all 8 benefits were zero, systematically. Group 2 produced 163 benefit estimates. Among these were 58 zero (=36%) and 12 (=7%) negative. There were again two questionnaires, where all 8 benefits were zero, systematically. These numbers are very high, so we must assume that the benefit estimates have a low validity. The main result of this experiment are not the risk and benefit estimates, but the participants´ judgements of which of the metrics are easy to estimate and whether they trust in their results

In the following, we discuss the validity of the effects of different treatments which have been observed.

Conclusion validity is concerned with the relationship of the treatment and the outcome, i.e. whether there is a statistically significant relationship between both. Thanks to a high number of participants, many effects observed have in fact been found to be statistically significant in most of the cases. So, we are certain that they were not accidentally caused. We enhanced this certainty by using several criteria for measuring what a "good" estimate is.

Internal validity is threatened if a relationship is observed between the treatment and the outcome, although there in fact is none. This may happen when the observed effect is caused by other factors of the experiment execution, which are not under investigation. We do not know of any other differences between the two groups than the different MUCs and requirements they had to treat, because they all got the same introductory presentation and information, had questionnaires which looked similar and did their estimates in the same room under the same conditions. And also, the diverse metrics were treated equally in the experiment. In Tables 1 and 2, we see that the profiles of both groups were almost equal

Construct validity refers to the extent to which the experiment setting actually reflects the construct under study, e.g. the ability of the measure chosen. To reduce such difficulties, we chose several metrics to measure what a "good" estimate is. Most effects were observed for all of these metrics.

External validity is associated with generalization. If there is a causal relationship between the construct of the cause and the effect, can the result of the study be generalized beyond the scope of our study? We believe that our estimators with their average of 15 years of working experience in computing are competent and therefore representative for experts which would be invited to a risk estimation workshop. However, they had to work under worst-case conditions with respect to information available to them: In practice, when estimating risk, the estimators will probably get some support for their task like historical data from previous experience in their company or public statistics. This helps them to get an idea of the order of magnitude in which the probability would be. Our estimators had to rely on their own experience solely. This situation is hopefully not found in practice. Nevertheless, we believe that while this situation led to estimates which are probably on the wrong order of magnitude, the effects which different treatments showed in this experiment would also be observed in practice. That means: Which of the risks is easier to estimate in the experiment, also is easier to estimate in practice.

Due to practical constraints (time limit), we had to use a constrained number of risks only, which might not be representative or typical for real-world estimation workshops.

Summarizing: We do not believe that the probabilities and damages estimated are correct, but maybe in the right order, that means more probable and more detrimental risks have been identified to be so. We expect that the effects of different treatments which we observed in our experiment would also be observed in practice. These effects have been found using several quality metrics and are partly statistically significant.

7. Summary

This paper describes a risk estimation experiment where IT-related risks are estimated by 50 participants. Like previous experiments, this one shows that risk estimation is difficult. The estimated probabilities and damages vary a lot: plus/minus two points on a scale of 1-7 points. The probabilities of MUCs 1 and 2 – for which statistical values are known – could approximately be estimated by the participants. They also agreed on the priorities of most of the MUCs and requirements, in terms of which is more important than the other. Although in this experiment we had participants with a high working experience, they did not feel more certain about their risk estimates than the students in our student experiments did before. However, our research question was not whether risk estimation is difficult or correct, but we wanted to compare different treatments, here: Different wording of MUCs and requirements, and different metrics used for probability and damage estimation.

We found clear answers to our three research questions, and these answers help to understand better what risk estimators can do and what they need. The research questions and answers were:

1. **Fuzziness: Are the estimates achieved for quantified Misuse Cases and requirements better than those for fuzzy or abstract ones?** The group with the quantified MUCs and requirements in fact felt better about their estimates than the group with the fuzzy wording - with respect to all criteria used: They feel more certain or less uncertain about their estimates, think

that estimation is less difficult, and expect more that their estimates are reasonable, realistic and useful.

2. **Metrics: Are damage and probability of risks easier to estimate on a scale of points than in percent or Euro?** Probabilities were easier to estimate on a point scale then in times per year or in %. For the damage estimation, the best scale was dissatisfaction (measured in points). Second was to estimate damage in points, then damage in hours of time lost, and most uncertain the participants believed their estimates in € to be.
3. **Predictability of uncertainty: Can estimators judge or even quantify the certainty of their estimates?** The estimators could predict which estimates (in terms of metrics) are more certain. However, they could no quantify these uncertainties.

So, from this experiment we learn: Clearer wording leads to better risk estimates, and relative probability and damage on point-scales are easier to estimate than absolute values in €, hours, times per year or %. Subjective dissatisfaction is easier to estimate than objective damage. Estimators can judge which of their estimates are more certain, but they can not quantify this uncertainty.

Acknowledgements. I thank the reviewers and empirical program chair for giving me the occasion to execute the experiment at REFS 2011, and also thank the REFSQ organization team for their great organizational support. A special thank also goes to the 50 participants who spent their time on estimating risk – what is a hard task.

References

1. Arora, A., Hall, D., Pinto, C.A., Ramsey, D., Telang, R. 2004. An ounce of prevention vs. a pound of cure: How can we measure the value of IT security solutions? Carnegie Mellon CyLab.
2. Berander, P. 2004. Prioritization of Stakeholder Needs in Software Engineering. Understanding and Evaluation. Licenciate Thesis, Blekinge Institute of Technology, Sweden, Licentiate Series No 2004:12.
3. Boehm, B.W., Fairley, R.E. 2000. Software Estimation Perspectives, IEEE Software 17(6) 22-26.
4. BSI (Bundesamt für Sicherheit in der Informationstechnik), 2005. BSI-Standard 100-3 Risk Analysis based on IT-Grundschutz Version 2.0, Germany, https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/ITGrundschutzstandards/standard_1003.pdf?blob=publicationFile (last visit: 6 March 2011)
5. Feather, M.S., Cornford, S.L. 2003. Quantitative risk-based requirements reasoning. Requirements Eng. J. 8(4): 248-265.
6. Feather, M.S., Cornford, S.L., Larson, T. 2000. Combining the best attributes of qualitative and quantitative risk management tool support. Proc. 15th IEEE Int. Conf. on automated software eng., Grenoble, France, 11–15 September 2000. IEEE Computer Society, 309–312.
7. Feather, M.S., Cornford, S.L., Gibbel, M. 2000. Scalable mechanisms for requirements interaction management. IEEE Int. Conf. on Requirements Eng., Schaumburg, USA, pp. 119-129.
8. Feather, M.S., Cornford, S.L., Kiper, J.D., and Menzies, T. 2006. Experiences using Visualization Techniques to Present Requirements, Risks to Them, and Options for Risk Mitigation. Proc. Int. Workshop on Requirements Eng. Visualization, Minneapolis/ St. Paul, Minnesota.

9. Gigerenzer, G. 2007. Das Einmaleins der Skepsis - Über den richtigen Umgang mit Zahlen und Risiken. Berliner Taschenbuchverlag, 3rd edition
10. Herrmann, A. 2011. Information Need of IT Risk Estimation – Qualitative Results from Experiments. Workshop "Requirements Prioritization for customer-oriented Software Development" (RePriCo'11), March 31, 2011, Essen, Germany, at REFSQ 2011
11. Herrmann, A., Paech, B. 2008. MOQARE: Misuse-oriented Quality Requirements Engineering. Requirements Engineering Journal 13(1) 73-86.
12. Herrmann, A., Paech, B. 2009. Practical Challenges of Requirements Prioritization Based on Risk Estimation. Journal of Empirical Software Engineering 14(6) 644-674.
13. ISO (International Standards Organization). 2002. ISO, Risk management – Vocabulary – Guidelines for use in standards, ISO Guide 73. Geneva: International Standards Organization
14. Karlsson, J. 1996. Software requirements prioritising. Proc. 2nd Int. Conf. Requirements Eng., 110-116.
15. Mayer, N., Rifaut, A., Dubois, E. 2005. Towards a Risk-Based Security Requirements Engineering Framework. Proc. 11th Int. Workshop on Requirements Eng. for Software Quality, Foundations of Software Quality REFSQ, Essener Informatik Beiträge, Band 10: 89-104.
16. Park, J., Port, D., Boehm, B., In, H. 1999. Supporting Distributed Collaborative Prioritization for WinWin Requirements Capture and Negotiations. Proc. Int. 3rd World Multiconference on Systemics, Cybernetics and Informatics SCI´99, Vol.2, 578-584.
17. Pearson, K. 1900. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. Philosophical Magazine, Series 5 50 (302) 157–175.
18. Raiffa, H., Richardson, J., Metcalfe, D. 2002. Negotiation analysis - the science and art of collaborative decision making. Cambridge: Belknap.
19. Sands, A., Tseng, V. 2009. SquareTrade - 1 in 3 Laptops fail over 3 years. http://www.squaretrade.com/pages/laptop-reliability-1109/
20. Schapira, M.M., Nattinger, A.B.,Colleen, A. McHorney. Frequency or Probability? A Qualitative Study of Risk Communication Formats Used in Health Care. Med Decis Making 21, 459.
21. Sindre, G., Opdahl, A.L. 2000. Eliciting Security Requirements by Misuse Cases. Proc. TOOLS Pacific 2000: 120-131.
22. Sindre, G., Opdahl, A.L. 2001. Templates for Misuse Case Description. Proc. 7th Int. Workshop on Requirements Eng.: Foundation of Software Quality – REFSQ, Essener Informatik Beiträge Band 6. Essen, Germany, 125-136.
23. Tversky, A., Kahneman, D. 1974. Judgment under uncertainty: Heuristics and biases. Science 185: 1124–1131.
24. Welty, G. 1972. Problems of Selecting Experts for Delphi Exercises. Academy of Management Journal 15(1) 121-124.
25. Xie, N., Mead, N. R., Chen, P., Dean, M., Lopez, L., Ojoko-Adams, D., and Osman, H. 2004. SQUARE Project: Cost/Benefit Analysis Framework for Information Security Improvement Projects in Small Companies. Software Engineering Institute, Carnegie Mellon University, Technical Note CMU/SEI-2004-TN-045

Part III

REFSQ 2011 Doctoral Symposium Proceedings

8 Doctoral Symposium

Editors

Vincenzo Gervasi
University of Pisa, Italy, gervasi@di.unipi.it

Barbara Paech
University of Heidelberg, Germany, paech@informatik.uni-heidelberg.de

Technical Programme

Report on the First REFSQ Doctoral Symposium

Vincenzo Gervasi[1] and Barbara Paech[2]

[1] Università di Pisa, Italy
[2] University of Heidelberg, Germany

Report from the Organizers

For the first time in the REFSQ series history, the 2011 edition hosted a Doctoral Symposium for Ph.D. students working in the general area of Requirements Engineering. The symposium was jointly organized by the authors, and intended to bring together Ph.D. students with the double purpose of encouraging networking and the establishment of links at an early stage in their careers, and of providing valuable advice from a panel of senior researchers on how to best bring the Ph.D. work to fruitful (and timely!) completion.

In response to the Call for Papers, six submissions were received from as many Ph.D. students, each consisting of two elements:

1. a research abstract describing the problem addressed, its relevance for research and practice, an outline of the intended solution, some consideration on the novelty of the proposed solution, and an outline of the research method applied and of the current stage of the work;
2. a recommendation letter by one of the supervisors, establishing relevance for Requirements Engineering research and presenting a general overview on the student's progress.

All proposals were independently reviewed by three members of the Symposium's Program Committee, who provided recommendation for acceptance or rejection. Where needed, a second phase of online discussion among the reviewers helped clarify differences in assessment, and to reach a consensus decision. As a result of this process, five out of the six submissions were accepted for presentation. One proposal was considered to be at too early a stage to be discussed at the Symposium, and the Committee opted to provide early feedback through pointers to relevant scientific literature instead.

In line with the REFSQ tradition, and with the overall goal, the Symposium was planned for special emphasis on discussions and interaction, rather than on presentations. To further encourage discussion not only with panelists and other participants to the Symposium itself, but also with the general audience of REFSQ, authors were invited to prepare posters to be displayed in the common area during the main REFSQ conference (so that participants to both the scientific and the industrial track of the main conference could examine them).

On the 31st of March, 2011, the Doctoral Symposium was opened on the premises of *paluno*, the Ruhr Institute for Software Technology of the University of Duisburg-Essen. Attendants included the five presenters, a group of senior researchers serving as advisors, and a student who had not submitted a proposal, but who was attending the Symposium in preparation for future developments of his work. Each presenter was allocated 50 minutes, with 20 minutes for presentation and 30 for discussion. As is typical of the REFSQ spirit, the discussions were very lively, both by the panelists and by fellow students, and session chairs often had to call a timeout to ensure that the whole symposium would run on schedule. Minutes of the various discussions were recorded during the symposium by both organizers, and (after some editing and clarification) provided to the presenters; this ensured that the students could benefit from a complete and clean trace of the whole discussion.

In addition to the presentations, the Symposium hosted a micro-tutorial offered by Dan Berry about how to complete a Ph.D. on time — reinforcing the main message of focusing on a well-defined problem before venturing into extending a proposed solution to loosely related issues. After the micro-tutorial, a brief session was devoted to collecting feedback and impressions from the students and panelists, with the purpose of improving the process for next year's edition.

After the Symposium, the five presenters were invited to submit a revised version of their research abstracts, taking into account the advice and suggestions received, as well as any further progress in their work that might have happened since the original submission. The reader will find these extended abstracts in the following pages.

It is our pleasure to, first of all, congratulate the presenting students and their supervisors on the quality of their presentations. While several of the proposals had, according to the advisors' opinions, over-ambitious goals (not an uncommon case for Ph.D. proposals!), all of them were about interesting problems and proposed well-argued solutions.

Second, we would like to thank the members of the REFSQ Doctoral Symposium Program Committee, who helped in the selection process and provided initial feedback to the students, and the panelists who participated in the lively discussions.

Last, but not least, we gratefully acknowledge the excellent logistic support we received from the local organizers, and notably the assistance of Vanessa Stricker and Philipp Schmidt.

Doctoral Symposium Organization

Program Committee

- *Vincenzo Gervasi*, University of Pisa, Italy (co-chair)
- *Barbara Paech*, University of Heidelberg, Germany (co-chair)
- *Dan Berry*, University of Waterloo, Canada
- *Sjaak Brinkkemper*, University of Utrecht, Netherland
- *Tony Gorschek*, Blekinge Institute of Technology, Sweden
- *Camille Salinesi*, University Paris 1 Panthéon – Sorbonne, France

Panelists

- *Dan Berry*, University of Waterloo, Canada
- *Jorg Doerr*, Fraunhofer Institute Experimental Software Engineering, Germany
- *Vincenzo Gervasi*, University of Pisa, Italy (co-chair)
- *Leah Goldin*, Shenkar College of Engineering and Design, Israel
- *Erik Kamsties*, University of applied Science and Arts Dortmund, Germany
- *Barbara Paech*, University of Heidelberg, Germany (co-chair)
- *Camille Salinesi*, University Paris 1 Panthéon – Sorbonne, France
- *Norbert Seyff*, University of Zürich, Switzerland

Local Organization & Proceedings

- *Vanessa Stricker*, paluno, Germany
- *Philipp Schmidt*, paluno, Germany
- *Wilhelm Springer*, University of Heidelberg, Germany

Presenters

- *Morayo Adedjouma*, CEA LIST, France: "A Model-Based Requirements Engineering Framework in an Automotive Certification Purpose" (supervisor: François Terrier)
- *Veit Hoffman*, RWTH Aachen, Germany: "Towards an Integrated Tool Supported Use Case Engineering Approach" (supervisor: Horst Lichter)
- *Ali Niknafs*, University of Waterloo, Canada: "The Impact of Domain Knowledge on the Effectiveness of Requirements Engineering Activities" (supervisor: Dan Berry)
- *Maria Pinto-Albuquerque*, Lancaster University, UK: "Jigsaw Puzzle Metaphor to Handle Imperfection in Requirements" (supervisor: Awais Rashid)
- *Kevin Vlaanderen*, Utrecht University, The Netherlands: "Process Improvement in Software Product Management: The Online Method Engine" (supervisors: Sjaak Brinkkemper and Inge van de Weerd).

A Model-Based Requirements Engineering Framework in an Automotive Certification Purpose

Morayo ADEDJOUMA*,**

* DELPHI Electronics & Safety 95972 ROISSY CDG Cedex
** CEA, LIST, Point Courrier 94, Gif-sur-Yvette, F-91191 France
Morayo.Adedjouma@{delphi.com;cea.fr}

Abstract. Over the years, critical systems are submitted to more and more constraints (functional, real-time, safety…). Given this, various requirements engineering (RE) techniques had developed and had reached a certain level of maturity: model-driven requirements engineering is one of them. Therefore, today, systems are defined according complex processes based on experience and business expectations. Nevertheless, this ad-hoc modeling does not yet match the normative expectations. Our work is performed in the general context of a model-based RE process with the specific focus to take into account the needs defined in standards for a certification support; more particularly, needs concerning the automotive domain with the ISO26262 standard.

Keywords: Requirements engineering, UML modeling, traceability, certification support, complex systems, automotive standards

1 Introduction

Electronic devices progressively increase in application domains where safety levels can be high. Any failure, software or hardware, can lead to serious loss of life, security and / or material. Thus, safety is not only a desirable quality for critical systems, but a required compliance to international regulations. It becomes paramount to have means which ensure the development of such secure software. In the automotive context, the upcoming ISO26262 standard [8] intends to frame the whys and wherefores (ins and outs) of such expectations. Indeed, the standard focuses on the assessment of functional safety proposing a system classification with ASIL (Automotive Safety Integrity Levels) levels, additional processes, activities, techniques and methods, expected output data through an application model and framework to illustrate the competence for managing systems. In parallel, embedded systems are defined according to complex processes combining different formalisms or methods from the specification steps until the code production. Among them, graphic representations have the benefit to help easily understanding a complex problem and communicating a system's functional and the requirements. Examples include the Unified Modeling Language (UML) and its profiles and we generally talk about model-driven engineering [7] in the framework of definition of these systems. The critical point is that these processes are defined without any regard on certification standards, as they are in most cases empirically constructed over business realities and existing systems. My research goal is to provide a consistent methodology that allows combining modeling with UML, requirements management [14] at models level and consideration of the ISO26262 standard.

This paper is organized as follows: section 2 briefly presents the current practices in automotive domain and different trends about the topic. Section 3 presents some related works. Section 4 presents our approach, the first results and possible next steps. Finally, we conclude and summarize this paper in section 5.

2 Current Practices

Major actors in the automotive domain (we can cite some European CESAR[1] project partners) use various efficient tools for the development and the management of their requirements like DOORS[2] or Reqtify[3]. In other hand, they use others tools to define the architectural representations with SysML [11] for example, an UML profile for modeling systems with as particularity to manage the effective specification of requirements and the control models with Matlab Simulink/Stateflow[4] of theirs applications. However, in this practice, the different activities are completely unrelated to each other because they are performed with different tools; and the part of the traceability with architectural elements is still often hesitant or even nonexistent. However, complete RE must also allocate requirements on system design elements according to ISO26262 standard. The Automotive SPICE (Software Process Improvement and Capability Determination) [3], an international standard used in major worldwide automotive firms as a "*framework for the assessment of processes*" also adopts this statement. Another drawback is that, there are no specific activities or methods to manage the functional safety for proving the compliance to the ISO26262 standard; even though many companies are already trying to incorporate the related recommendations (as in CESAR project).

Incompleteness in the requirement management for respecting these standards is then evident since the traceability information is not ensured between requirements and design elements; furthermore, the safety assessment is not handled.

Better process support is therefore needed on the system level for explicitly managing the safety and engineering information related to the overall requirement and architecture definition in a formal representation.

[1] CESAR Project, www.cesarproject.eu/
[2] IBM Rational DOORS, www.ibm.com
[3] Reqtify, www.geensoft.com/en/article/reqtify/
[4] MATLAB, www.mathworks.fr

3 Related Works

Recent modeling improvements and extensions have been made with the SysML UML-profile [4] for a better consideration of the requirements: for instance the DARWIN4Req approach [5] and EAST-ADL2 profile [6].

DARWIN4Req is a specific SysML-profile strictly focused on requirement management and their traceability. As this profile does not consider the system architecture, its usage must be joined with other specific profiles and languages dedicated to system architecture. Furthermore, it does not take into account the ISO26262 recommendations for the requirements management in a certification aim.

EAST-ADL2 is an initiative from the automotive domain, which presents the modeling infrastructure, i.e. how the modeling elements should be represented in UML; it requires system descriptions on several abstraction levels, from top-level user features to tasks and communication frames. It involves the expression of non-structural aspects of the system like requirements based on SysML concepts. One of these advantages is the connection with the de-facto automotive standard AUTOSAR[5]. Although EAST-ADL2 recent version integrates some ISO26262 requirements like the SafetyGoal concept for instance, it does not take into account the necessary quality characteristics of requirements asked by the ISO26262. One example is the requirement “status” which can have the values "*proposed, assumed, agreed or reviewed*" and which does not appear.

As other common weakness, these technical solutions are hardly associable with different RE techniques and tools (DOORS and Reqtify) that may exist and that offer very rich mechanisms of elicitation, refinement or decomposition of requirements and that may be based on dependability techniques or fault management for example. The Requirements Interchange Format, namely ReqIF [12], can address this issue. It is a standardized open, generic, non-proprietary and tool-independent format to support requirements exchange processes between different requirements management tools. Requirement information is exchanged by XML documents without losing the advantage of requirements management at the authoring tools borders.

Some research and industrial initiatives (in CESAR project for example) for assessing RE are also currently being developed in model-driven community, each focusing on part of our problem.

4 Research Ideas and Methodology

The theory is organized around the followings topics: 1) the identification in the ISO26262 Standard of different needs to be covered for requirements and their means of production and the consideration through and in models; 2) the definition of requirements traceability mechanisms (trace to requirements, to model elements and to a proposed system architecture) and multi-formalisms modeling through interoperability tools; and finally 3) the definition of a comprehensive methodology to guide the design of systems in such environment.

[5] AUTOSAR, www.autosar.org

As exposed in the previous section, some contributions were very close to our approach, but the bibliography denotes that no current work is able to tackle every part of our problem. We draw our inspiration from these different approaches and researches for building a solution adapted to the automotive needs.

4.1 Model-based Requirement Engineering

Our Requirement Engineering Management accordIng to Automotive Standards (REMIAS) profile [1] (see Fig. 1) aims to implement in models the normative recommendations of the field for requirements engineering, especially those of ISO26262 standard. Instead trying to redefine a new profile, and thanks to the profile capabilities [13], we choose to reuse, integrate and complete two complementary profiles: DARWIN4Req for the requirement management and EAST-ADL2 for the system architecture design.

We extend DARWIN4Req by adding the necessary properties, summarized in the "Specification and management of safety requirements" chapter from ISO26262-8 chap. 6. The DARWIN4Req requirement classification is not compliant with the hierarchical and organizational structure imposed by the ISO26262. Indeed, DARWIN4Req classifies "safety requirements" as quality requirements while in the ISO26262, any requirement (even a functional one) can be a "safety requirement". We change it to state that any requirement can be a "safety requirement".

EAST-ADL2 proposes a good way to specify automotive architecture systems. However, the architecture definition in several abstraction levels is hard to follow for engineers. To simplify the process, our goal is to propose a mechanism which, from a system architecture level (functional analysis architecture in EAST-ADL2) to automatically generate for the users the detailed associated design (software design architecture and hardware design architecture in EAST-ADL2).The REMIAS profile will be implemented as a static profile. Thus, some requirements properties and hierarchical structure will be automatically inherited. For example, the ASIL (Automotive Safety Integrity Level) between a requirement and his derived.

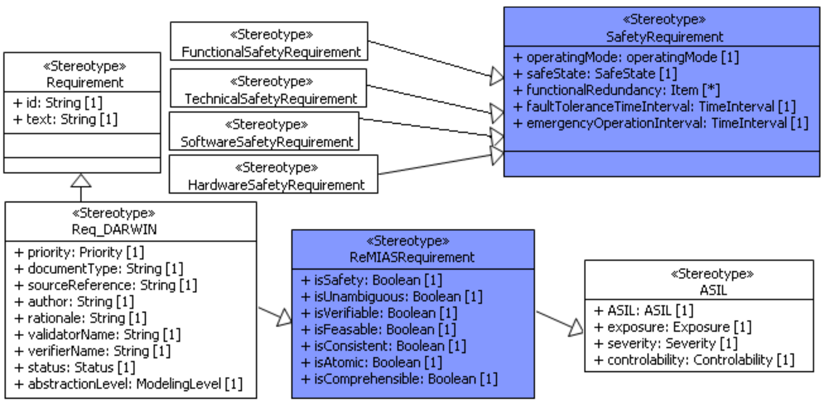


Fig. 1. REMIAS requirement metamodel

4.2 Multiformalism Requirements and Interoperability Tools

In automotive industry, requirements engineers work with requirement management tools like DOORS and Reqtify. Unfortunately, these ones do not manage elements of system architecture, done in modeling environments. Therefore, the challenge is to allow to import and export the requirements list from native customer specification documents without writing them manually as a model in a modeler. We will use ReqIF to perform this task.

Commercial tools like DOORS and Reqtify have already incorporated an importer/exporter for RIF (old version of the ReqIF standard). EAST-ADL2 also proposes a RIF extension. But, since RIF format file exported by these one are different (implemented from different metamodels), the tools cannot access the exported data of the other, making the interoperability impossible. Furthermore, there are some differences between the RIF and the ReqIF standard.

In our approach, the requirements exchange is performed between simple MS documents and a SysML Requirements model into Papyrus MDT[6] tool using a mapping between SysML concepts and ReqIF concepts (see Fig. 2). For this purpose, we had defined a ReqIF Ecore metamodel with EMF (Eclipse Modeling Framework). The metamodel allows defining SysML requirements model, fitting one of ReqIF concepts. Then, we use ReqIF XML schema metamodel in order to apply it in our MS document and export data using this template. This one is performed by using both XML complements and Developer extensions from Microsoft.

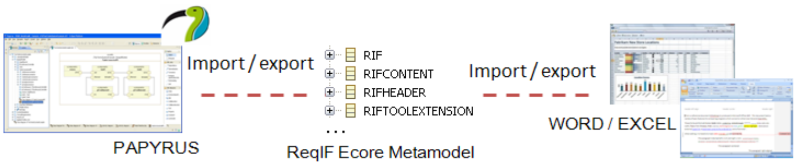


Fig. 2. Requirements exchange using ReqIF from MS documents to MDT Papyrus tool

The XML data file exported from MS documents is compliant to the ReqIF metamodel. Papyrus MDT tool can use it to define a SysML requirement model. Then, the transformation is performed by using the ATL[7] language, a model transformation technology (this result has been accepted for publication and it will appear in [2]). To further support the ISO26262 automotive standard, the requirements attributes will be extended, based on the REMIAS profile.

Another interesting goal is to animate the UML architecture model with functional MATLAB Simulink/Stateflow models. UML is a graphical language, which allows making static representations. It does not include mechanisms for simulation. However, in automotive industry, generally, Simulink models describe the behavior of applications. EAST-ADL2 yet proposes to reference Simulink modules in the behavior of components. We wish go beyond: in order to obtain an early evaluation of

[6] Papyrus MDT, wiki.eclipse.org/MDT/Papyrus

[7] ATL, www.eclipse.org/atl/

architectural choices at system level, we will animate UML models with data resulting from Simulink/Stateflow models simulation. For that, we need first to define a system architecture modeling approach (typing, naming…) and some others extensions to allow interaction with Simulink/Stateflow models.

4.3 Methodology Support

For introducing the certification point in our methodology, we have presented the ISO26262 standard in a conceptual process model (requirements, methods…) [1] and his mapping by activity or sub process on the different part of the engineering development process (Fig. 3).

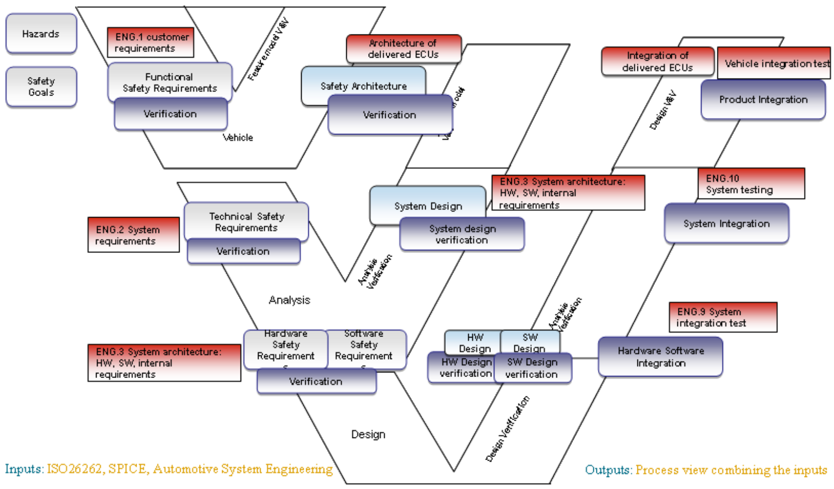


Fig. 3. Conceptual process model

The aim is to ensure the traceability between system requirements from a development process and requirements from standards; and the traceability between activity/process and system artifacts for a project monitoring purpose. The next step is to formalize the latter with a process modeling language like SPEM [10] or BPMN [9]. By adding an action language, we hope to parameterize in order to obtain a specific process to develop a given product: according to ASIL, tools, methods, users…

5 Conclusion

This PhD thesis work deals with two major domains: requirements engineering and safety management in the ISO26262 standard framework.

Our proposal is a UML-based approach that provides advanced techniques and notations supporting the full requirements life cycle, from their definition until their architectural allocation for E/E automotive embedded systems. The REMIAS profile integrates some recommendations of automotive standards, respectively from, among the most important, the "Specification and management of safety requirements" chapter 6 of ISO 26262-8 and the 3 first Automotive SPICE Engineering Process Groups (called ENG): requirements elicitation (ENG1), system requirements analysis (ENG2), system architecture design process (ENG3). Our ReqIF export and import a requirement repository into UML-based modeling tool from MS requirements specification documents, avoiding redundant working for requirement engineers. A study of standards is also performed to extract safety concepts and processes rules relevant to automotive field for a methodology support.

The research work is validated on empirical systems that we built incrementally, i.e. we enrich it as we progress with our theory. These are real embedded automotive systems projects currently in development. Therefore, the feedbacks are useful to revise and improve the methodology. To mitigate the threat to external validity, we plan to define such measurements to assess the pertinence and usefulness of our approach.

References

1. Adedjouma, M., Dubois, H., Maaziz, K., Terrier, F.: A Model-Driven Requirement Engineering Process Compliant with Automotive Domain Standards. In: 3rd MDTPI Workshop on Model-Driven Tool & Process Integration (2010)
2. Adedjouma, A., Dubois, H., Terrier, F.: Requirements Exchange: from Specification Documents to Models. In: 6th IEEE International workshop UML and AADL, USA (2011)
3. Automotive SIG: Automotive SPICE, Process Assessment Model. Version 2.4, Status: Released 2008-08-01 (2008)
4. Dos Santos Soares, M., Vanckren, J.: Model-Driven User Requirements Specification using SysML. Journal of Software vol. 3, no. 6 (2008)
5. Dubois, H., Peraldi-Frati, M.-A., Lakhal, F.: A model for requirements traceability in a heterogeneous model-based design process. In: 15th IEEE International Conference on Engineering of Complex Computer Systems, pp. 233--244, UK (2010)
6. EU Project ATESST: EAST-ADL2 profile Specification, the ATESST Architecture Description Language. Version 2.1 (2010)
7. Gérard, S. Babau, J-P, Champeau, J.: Model Driven Engineering for Distributed Real-time Embedded Systems. Lavoisier (2005)
8. International Organization for Standardization: ISO Working Draft international standard IS0/DIS 26262. Baseline 12 (2009)
9. OMG: OMG Business Process Model and notation. Version 1.2, OMG document number: formal/ 2009-01-03 (2009)
10. OMG: OMG Software & Systems process Engineering MetaModel. Version 2.0, OMG document number: formal/ 2008-04-01 (2008)
11. OMG: OMG System Modeling Language (OMG SysML). Version 1.2, OMG document number: formal/ 2010-06-01 (2010)
12. OMG : Requirements Interchange Format (ReqIF). Version Beta 1.0, OMG document number: dtc/2010-07-01 (2010)

13. Selic, B.: A Systematic Approach to Domain-Specific Language Design Using UML. In: 10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC'07), pp.2-9, Greece (2007)
14. van Lamsweerde, A.: Requirements Engineering, From System Goals to UML Models to Software Specifications. Wiley (2009)

Towards an Integrated Tool Supported Use Case Engineering Approach

Veit Hoffmann and Horst Lichter

Research Group Software Construction
RWTH Aachen University
Ahornstr.55, 52074 Aachen, Germany
{vhoff,lichter}@swc.rwth-aachen.de

Abstract. Use cases have become a widespread approach for capturing requirements, yet they still face some major acceptance issues. Current use case modeling techniques lack a systematical engineering approach and adequate tool support. Today especially conclusive quality assurance measurements are missing. In this article we describe a research project which aims to overcome current methodical and technical problems of use case modeling by introducing a tool supported engineering approach for text based use case descriptions that includes different quality assurance measures.

Keywords: Use case, quality assurance, engineering, tool support

1 Introduction

Empirical studies [27] show that since their introduction by Ivar Jacobson in 1986 [16] Use Cases (UC) have become one of the most important requirements techniques to specify functional behavior. UC modeling nowadays is an integral part of most modern development processes like the Unified Process[17].

According to [1, 4, 7] high quality UCs are a major driver for the whole software development process. They are a central means for communication between project stakeholders (e.g. users, customers, management, designers, and testers), which is crucial for building the right system. Additionally UCs link requirements to other software artifacts such as design, implementation, and test cases. Yet despite the importance of UCs for the whole development process and the severe problems that might results from low quality [24] current UC modeling approaches provides only insufficient guidance to systematically develop high-quality UCs.

The remainder of this paper is structured as follows. First we will present the addressed research questions and situate our work against related work. Then we will describe general considerations on a UC development process and its impact to an engineering approach. Afterwards we will present a brief overview of our UC engineering approach. Finally we will briefly sketch our contribution so far alongside some concluding remarks.

2 Research Questions

In order to indentify the main problems of the application of UC techniques in practice, we have analyzed the state-of-practice based on a comprehensive literature review and a survey we conducted in 2009 [27] where we got information from more than 70 software development companies. In summary practitioners today identify three mayor shortcomings of UC techniques:
1. They lack constructive guidance during UC creation.
2. They lack reliable indicators for UC quality and meaningful means to assess those indicators correctly and quickly.
3. They lack sufficient tool support.
Based on these analyses and the results of other studies [10] we have formulated three main research questions:
Q1: What is an appropriate process to develop high quality UC models?
Q2: What are characteristics of high quality UC models and how can we evaluate their quality?
Q3: Which kinds of tools are needed to support high quality UC development?

Hence, the overall goal is to provide a systematic tool supported UC engineering approach. To reach this goal we apply a tool-prototype driven approach where each conceptual research result is developed in parallel with a tool prototype. Thus we are able to validate our results early by applying them in small academic projects (e.g. master theses) as well as in medium size industrial projects.

3 Related Work

Current state-of-practice UC development methods [1, 7, 19] don't cover quality assurance sufficiently. Although they mention constructive guidelines and manual reviews to ensure UC quality they don't discuss when or how to apply them and completely omit other quality assurance techniques.

Several other approaches explicitly address UC quality assurance. Most common are constructive guidelines [9] and manual inspection checklists [8]. Besides, semi-automated heuristics or metrics either based on the UCs structure [3, 13, 11] or on linguistic analyses [12, 6] are discussed. Moreover, dynamic animation approaches [22, 23, 28] have been researched. All these publication address only specific aspects of quality assurance yet they are developed as if they were applied as the only quality assurance technique and don't give hints how they could be integrated in a systematic approach. Drenger et. al. [10] on the contrary present a method that integrates several quality assurance measures to address all different kinds of UC defects. However this method is mainly focuses on fully qualified UC descriptions, where we aim for an approach that is especially able to asses UC specifications in early stages.

Today several different commercial [5, 21, 26] and non commercial [25, 18, 22] UC tools are available, some which even integrate some quality assurance measures. Yet we are not aware of any tool that covers a whole UC driven requirements engineering process and integrates different quality assurance means to a UC engineering environment.

4 General Considerations on a UC Development Process

Unlike most other techniques UCs aren't utilized for just one specific purpose. Instead they address different aspects of requirements engineering and support several other downstream development tasks. Consequently, UCs typically have a special lifecycle and they are developed in a dedicated sub process consisting of several design phases [1, 4, 7]. In this process the UCs evolve in a breadth first manner incrementally and pass consecutive abstraction levels each addressing one or more specific project risks. I.e. short informal UC sketches addressing the project scope grow into detailed structured descriptions of the system behavior in context that support software testing and design. Besides, UC development is always influenced by a larger project context. The UC design process is adjusted depending on the tasks that should be addressed with UCs (e.g. testing or user documentation) and the UCs' format is adopted according to other information that must be connected with the UCs (e.g. domain glossary). As a consequence a UC engineering approach must include means to create UCs on all abstraction levels and to assess their appropriateness for a specific purpose as well as for the evolution to the next abstraction level. Moreover it must include a notation and a development method that are adoptable to different project contexts.

5 UCE: An Integrated Use Case Engineering Approach

UCE (Use Case Engineering) is an engineering approach that addresses UC development and quality assurance throughout all development stages. UCE can be tailored to different project contexts. Figure 1 depicts the core components of UCE: It is based on a common formally defined UC meta model, offers methodical guidelines, provides a UC specific quality assurance approach based on animation and metrics, and is supported by a set of dedicated tools. In the remainder of this section we briefly present UCE's core components.

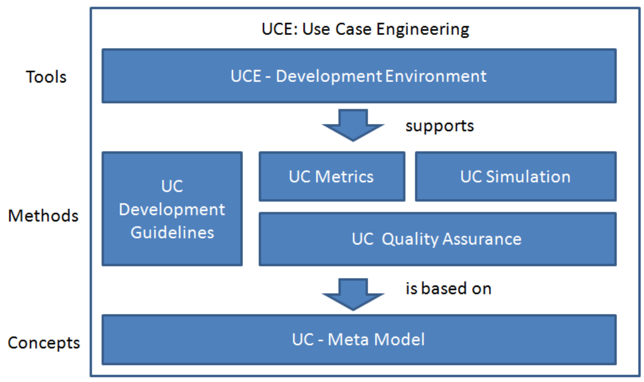


Fig. 1. UCE core components

5.1 UC Meta Model

The UC meta model is the central foundation for all other UCE components because is precisely defines all needed concepts in the context of UCE. This way it clearly constitues the semantics of UCs on all different abstraction levels. Thus it includes specific UC templates for all abstraction levels (e.g. initial, ...) proposed by current UC development methods and defines sanity constraints for the content of these templates. (e.g. that each extension point must be connected to at least one extend relation). Moreover the UC meta model offers a simple concept to integrate UCs with other requirements engineering artifacts (e.g. domain glossary) by providing special reference types.

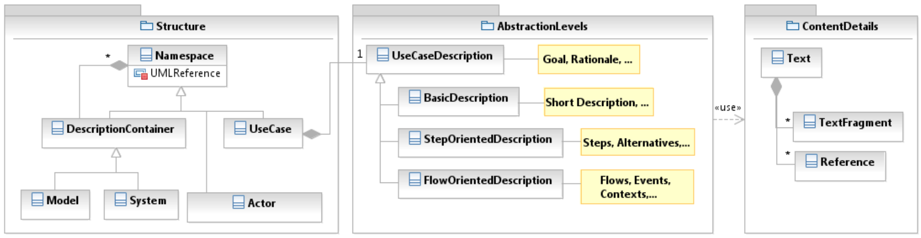


Fig. 2. UCM - Central concepts

5.2 Development Guidelines

Development guidelines are a proven good means to avoid errors that are difficult to discover by analytical quality assurance measures (e.g. completeness) [10]. UCE therefore provides a set of guidelines derived from collected best practices. They describe which modeling tasks should be performed to create UCs on a specific abstraction level, give hints about the task's central aspects and call to attention important pitfalls. Additionally the guidelines include quality metrics and point out when to use inspections for quality assurance and which aspects shall be reviewed.

Yet as we aim to address UCs in a variety of different project contexts we need a flexible guideline approach. Based on the observation that UCs are developed in an iterative breadth first manner, we want to develop a "UC development framework". In short this framework consists of a collection of guidelines for different modeling tasks together with a rule based approach to select those tasks that best fit to a concrete project setting by analyzing specific projects characteristics (like project size or criticality). Obviously this has great impact on the UCE development tools; they have to be adaptable to the applied UC modeling tasks.

5.3 UC Quality Assurance

In order to assess the quality of UC models important quality aspects must be defined first. Moreover, measurable indicators for those quality aspects must be identified to derive respective meaningful quality metrics. Our research has shown that it is impossible to define one single concise UC quality model, because UCs are utilized for very different purposes with completely different demands that must be considered when qualities are defined. Therefore we have applied a goal oriented approach based on the GQM [2] to identify usage specific quality metrics. Currently we have developed four different UC quality models to assess important quality aspects of the four main UC development stages (scope, planning, conceptual and detailed modeling) [20]. Moreover, we have identified a set of promising UC metrics for most quality aspects of those models. Yet our metrics still need to be tailored and evaluated in practice. During metric definition we observed that several quality aspects couldn't be assessed on the UC model alone (e.g., in order to assess understandability of a UC model the glossary defining the domain terms must be considered as well). Therefore we have extended those quality metrics with specific assumptions about the project context (e.g. other artifacts that must exist). Although these assumptions limit the applicability of the metrics we have evidence that they lead to more appropriate measurement results.

5.4 UC Animation

Despite the frequent usage of UC reviews in practice the structure of UC descriptions induces two principle problems: First, especially non technical stakeholders don't understand the specified behavior [14]. Second, the UC reviewers report problems to evaluate the system behavior as a whole [10]. Animation of UCs helps to overcome those problems, since it is a prototyping technique and provides an intuitive means to assess the behavior of a system. Stakeholders can experience the behavior by entering simple stimuli and thus evaluate the modeled functionality by stepping through the specified scenarios. Consequently, UCE offers a means to animate UC models, described in more detail in [14]. The basis for the animation is a dedicated UC execution model which can be generated even from partially incomplete UC models and therefore enables very early animation. On the other hand it precisely represents the behavior, since it is aware of the execution context. By means of UCE's animation environment we enable non technical stakeholders to assess the specified behavior, since no knowledge of the formal execution semantics is required. Moreover we have enriched the animation environment with GUI prototypes to offer an even more intuitive analysis of the behavior described in a UC model.

5.5 Nautilius: The UCE Development Environment

Alongside our conceptual research we are developing the UCE development environment called Nautilus [15] based on current Eclipse technologies. Nautilus is a UC development toolbox completely based on our UC meta model. It provides tools to easily create UC models on different abstraction levels and adds methodical guidance e.g. for their iterative evolution. Nautilus currently supports the measurement of UC metrics. Furthermore it provides a UC animation environment where the animations can be enriched with GUI prototypes to give the user an intuitive insight in the UC behavior. Additionally, all animation runs of a UC review can be stored for later analyses, like coverage measures.

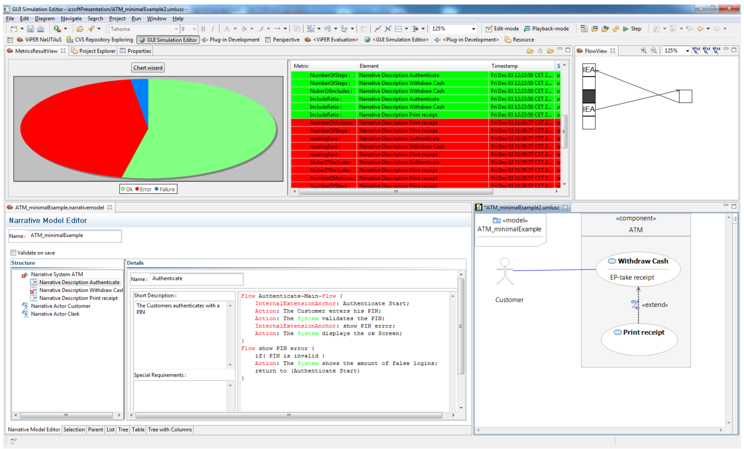


Fig. 3. Nautilius - UC development environment (Screenshot)

6 Contribution & Conclusion

As presented in this paper our overall research goal is to develop an integrated tool supported UC engineering approach. In 2008 we started our research with the introduction of the NarrativeModel (now UCM) a metamodel for detailed, flow-oriented, textual UCs alongside Nautilus an adjacent tool support for their creation and evaluation [15]. Since then we have extended Nautilus with a UC animation environment and a technical environment for the evaluation of UC metrics. In the future we will add different dynamic critiques and markups based on those metrics as proposed by Knaus et al. [18] and possibly including heuristic guidance for defect removal. Currently we are tailoring a set of goal oriented candidate UC quality metrics we derived by a goal oriented approach, we will present in near future. Moreover we are currently developing concepts for a method driven online help and its integration in Nautilus. Finally we are currently evaluating UCM and its tool support in several case studies with students and in small cooperation projects with an industry partner.

References

1. Armour, F., Miller, G.: Advanced use case modeling: software systems. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (2001)
2. Basili, V.R.: Software modeling and measurement: the goal/question/metric paradigm. Tech. rep., College Park, MD, USA (1992)
3. Bernardez, B., Duran, A., Genero, M.: Metrics for Software Conceptual Models, chap. Metrics for Use Cases: A Survey of Current Proposals, pp. 59–98. Imperial College (2005)
4. Bittner, K., Spence, I.: Use Case Modeling. Addison-Wesley Longman, Amsterdam (September 2002)
5. CaseComplete. http://www.casecomplete.com/
6. Ciemniewska, A., Jurkiewicz, J., Olek, L., Nawrocki, J.: Supporting use-case reviews. In: Abramowicz, W. (ed.) Business Information Systems, Lecture Notes in Computer Science, vol. 4439, pp. 424–437 (2007)
7. Cockburn, A.: Writing Effective Use Cases. Addison-Wesley Professional (January 2000)
8. Cox, K., Aurum, A., Jeffery, R.: An experiment in inspecting the quality of use case descriptions (2008)
9. Cox, K., Phalp, K.: Comparing use case writing guidelines. In: 7 th Int. Workshop on Requirements Engineering: Foundation for Software Quality. pp. 101–112 (2001)
10. Denger, C., Paech, B.: An integrated quality assurance approach for use case based requirements. Modellierung pp. 59–74 (2004)
11. El-Attar, M., Miller, J.: Matching antipatterns to improve the quality of use case models. IEEE International Conference on Requirements Engineering pp. 99–108 (2006)
12. Fantechi, A., Gnesi, S., Lami, G., Maccari, A.: Applications of linguistic techniques for use case analysis. Requirements Engineering 8, 161–170 (2003), 10.1007/s00766-003-0174-0
13. Henderson-Sellers, B., Zowghi, D., Klemola, T., Parasuram, S.: Sizing use cases: How to create a standard metrical approach. In: Bellahsne, Z., Patel, D., Rolland, C. (eds.) Object-Oriented Information Systems, Lecture Notes in Computer Science, vol. 2425, pp. 109–114 (2002)
14. Hoffmann, V., Lichter, H.: A model based narrative use case simulation environment. Proceedings of the 5th International Conference on Software and Data Technologies (ICSOFT 2010) 2, 63–72 (2010)
15. Hoffmann, V., Lichter, H., Nyßen, A., Walter, A.: Towards the integration of uml- and textual use case modeling. Journal of Object Technology 8(3), 85–100 (May 2009)
16. Jacobson, I.: Object oriented development in an industrial environment. OOPSLA pp. 183–191 (1987)
17. Jacobson, I., Booch, G., Rumbaugh, J.: The Unified Software Development Process. Addison-Wesley Professional (February 1999)
18. Knauss, E., Lübke, D., Meyer, S.: Feedback-driven requirements engineering: The heuristic requirements assistant. In: ICSE. pp. 587–590 (2009)
19. Maiden, N., Jones, S.: The rescue requirements engineering process: An integrated user-centred requirements engineering process, version 4.1. Tech. rep. (Feb 2004)
20. Morys, R.: Metric based Quality Modeling of Use-Case Based Requirement Specications. Master's thesis, Reasearch Group Software Construction - RWTH-Aachen University (2010)

21. Rational Software Architect. http://www-306.ibm.com/software/awdtools/architect/swarchitect/
22. Some, S.S.: Supporting use case based requirements engineering. Information and Software Technology 48(1), 43 – 58 (2006)
23. Sutcliffe, A., Maiden, N., Minocha, S., Manuel, D.: Supporting scenario-based requirements engineering. Software Engineering, IEEE Transactions on 24(12), 1072 –1088 (Dec 1998)
24. The Standish Group: Chaos chronicles v3.0 technical report. Tech. rep., The Standish Group (2003)
25. Use Case Editor (UCEd). http://sourceforge.net/projects/uced/
26. Visual Paradigm for UML. http://www.visual-paradigm.com/product/vpuml/
27. Weidmann, C., Hoffmann, V., Lichter, H.: Einsatz und Nutzen von Use Cases - Ergebnisse einer empirischen Untersuchung. Softwaretechnik-Trends Band 29, Heft 2, ISSN 0720-8928 (May 2009)
28. Whittle, J., Jayaraman, P.K.: Generating hierarchical state machines from use case charts. In: Proceedings of the 14th IEEE International Requirements Engineering Conference. pp. 16–25. IEEE Computer Society, Washington, DC, USA (2006)

The Impact of Domain Knowledge on the Effectiveness of Requirements Engineering Activities

Ali Niknafs
David R. Cheriton School of Computer Science
University of Waterloo
aniknafs@cs.uwaterloo.ca

Supervisor: Daniel Berry

Abstract. It is believed that the effectiveness of software engineering activities largely depends on the individuals involved. There are various factors influencing individuals' effectiveness in different software engineering activities. One of the factors that has attracted a little research is knowledge of the problem being solved. While a software engineer's having in depth domain knowledge helps one to understand the problem easier, he/she can fall for tacit assumptions of the domain and might overlook some issue that seem obvious. We have gathered the limited literature on the topic from different areas including requirements engineering and business analysis. The hypothesis on which this research is based is that adding a domain ignorant to a requirements engineering team improves the effectiveness of the process. A series of studies including controlled experiments are planned to explore this hypothesis.

Keywords: Requirements Engineering, Empirical Software Engineering, Requirements Knowledge

1 Motivation

There is a consensus that the cornerstone of software development process is deciding precisely what to build [1]. The process of arriving at a set of features that are needed to be developed is referred to as requirements engineering (RE). RE still needs to be studied in depth since many of the typical software problems arise from deficiencies originated from this early stage of development. One of the issues with RE is the huge gap between what the customer wants and what the analysts think the customer wants. To overcome this gap, it has long been believed that analysts need to be experienced in the customer's domain to be productive when performing a RE activity. On the other hand, knowledge of the problem domain seems to have a counterproductive aspect. In other words, lack of domain knowledge might in fact have some benefits in RE activities. One of the potential benefits of this lack of knowledge is the one observed by Berry [2]; i.e. the ability of a domain ignorant to state his/her ideas regardless of any domain assumptions and ask revealing questions that could lead to unveiling issues that domain experts have overlooked.

Following the Goal-Question-Metric template [3], the goal of this research is to improve the effectiveness of RE process from the viewpoint of project managers, in the context of both laboratory projects and real-world projects. The major question of this research is, "How does one form the most effective team of requirements engineers for an activity involving knowledge about the domain of the computer-based system (CBS) whose requirements are being determined by the team?"

To answer this research question, we are going to investigate the factors influencing the effectiveness of a RE team as it goes about several activities of RE. We want to measure how knowledge of the domain of the CBS affects the effectiveness of a RE team doing domain-knowledge intensive activities. Also other factors like educational and industrial background will be taken into consideration. Creativity is another factor to be considered since it plays an important role in idea generation tasks such as brainstorming. Personality traits are other factors. Extroverts might be more effective in team activities than introverts. Finally, as a secondary question, we are going to see whether the knowledge of a similar domain to the domain under study impacts the effectiveness of an individual.

The main hypothesis coming from the research question stated above is:

Adding a domain ignorant to a requirements engineering team improves the effectiveness of any requirements engineering process the team performs.

And the according null hypothesis would be:

The presence of a domain ignorant in a requirements engineering team has no effect on the effectiveness of any requirements engineering process the team performs.

To investigate the main hypothesis, the following subhypotheses (denoted by H_{X1}) and corresponding null subhypotheses (denoted by H_{X0}) are defined:

- H_{11}: The effectiveness of an individual in a requirements elicitation or inspection activity is affected by his/her prior knowledge of the problem domain.
- H_{10}: The effectiveness of an individual in a requirements elicitation or inspection activity is not affected by his/her prior knowledge of the problem domain.
- H_{21}: The effectiveness of an individual in a requirements elicitation or inspection activity is affected by his/her requirements experience.
- H_{20}: The effectiveness of an individual in a requirements elicitation or inspection activity is not affected by his/her requirements experience.
- H_{31}: The effectiveness of an individual in a requirements elicitation or inspection activity is affected by his/her educational background.
- H_{30}: The effectiveness of an individual in a requirements elicitation or inspection activity is not affected by his/her educational background.
- H_{41}: The effectiveness of an individual in a requirements elicitation or inspection activity is affected by his/her experience with similar domains.
- H_{40}: The effectiveness of an individual in a requirements elicitation or inspection activity is not affected by his/her experience with similar domains.
- H_{51}: The effectiveness of an individual in a requirements elicitation activity is affected by his/her creativity level.
- H_{50}: The effectiveness of an individual in a requirements elicitation activity is not affected by his/her creativity level.

2 Related Works

Very few studies have investigated the impact of domain knowledge on software engineering activities. This section summarizes the relevant existing SE studies conducted in either academic or industrial settings.

Berry [2] made the same observation based on his own and his then wife's experience with RE projects. As he noted later [4], one of the earliest observations of the impact of ignorance comes from the Burkinshaw's statement during a conference in 1969 [5]:

> "*Get some intelligent ignoramus to read through your documentation and try the system; he will find many "holes" where essential information has been omitted. Unfortunately intelligent people don't stay ignorant too long, so ignorance becomes a rather precious resource. Suitable late entrants to the project are sometimes useful here.*"

One interesting observation was made by Dieste et al. [6] in a survey on requirements elicitation techniques. They concluded that a requirements analyst's experience with interviewing as an elicitation method and his/her experience with the problem domain do not affect the quantity of the ideas generated during an interview.

Kenzi et al. [7] have studied the effect of domain knowledge on conducting interviews and on the preferences of different elicitation techniques throughout the elicitation process. Consistent with Dieste et al. they claimed that those without domain knowledge can be effective in interviews. Although they have indicated that they have not explored the effect of an analyst's prior domain knowledge.

Ferrari et al. [8] studied the impact of requirements knowledge and experience on software architecture tasks. Their study suggests that architects with requirements knowledge and experience perform better than those without.

Carver et al. [9] have investigated the impact of educational background on the effectiveness of individuals in an inspection task. They have conducted a controlled experiment having two types of subjects; those who have studied computer science as their university major and those who have studied something else. What they have observed in this study is that the general knowledge of computer science did not improve the quality of the inspection and the individuals in non-computing majors did even better in detecting defects. While this study did not focus on the effect of problem domain knowledge of the subjects, their findings seem to be consistent with our hypothesis.

RE is similar to business analysis because many processes involved in these two areas are identical [10]. Each area involves the elicitation of information from end users. One of the sightings in business analysis is the one made by Miles Barker [11]:

> *"…and we shouldn't forget, that if we just stick to the same domain and the same group of insiders, innovation will be much harder to come by…People will usually come up with the same ideas that they have at their last domains.*"

3 Execution

This section describes how we are going to test the hypotheses.

3.1 Controlled Experiment

Controlled experiments are probably the most effective methods to validate a hypothesis. However, in practice, it is usually very difficult to foresee all the factors that are required to be taken into consideration. Therefore, we decided to perform a series of pilot studies to decide which variables to include in the plan for the actual experiment. Among many RE activities, we focused on elicitation and inspection of requirements, because they are doable in the normal confines of a controlled experiment. These activities are usually done in sessions each lasting 1 or 2 hours.

There are other activities that might need to be evaluated. For any activity in which implicit knowledge hurts the activity's effectiveness, ignorance might be beneficial. Architecture design, requirements prioritizing, quality control, writing test cases, and documenting requirements, could be such activities.

The aim of the experiment described in the following, is to study the effectiveness of various configurations of two kinds of subjects in an activity to find or inspect the requirements of a CBS. The two kinds of subjects are: those having prior experience in the domain under study, and those completely ignorant of the problem domain.

3.1.1 Variables

The only dependent variable in the controlled experiments is the effectiveness of the experiment task. For the elicitation task, the number of generated requirements and their value are considered to be the dependent variables while in the inspection task, the number of defects detected is the dependent variable.

Independent variables include educational background, industrial experience, requirements writing experience, requirements inspection experience, creativity level, and a few personality traits.

3.1.2 Subjects

Subjects will be drawn from academia and industry. We will welcome everybody with divergent levels of expertise and educational background.

3.1.3 Procedure

Each subject will be asked to fill out a questionnaire about his education level, field of study, prior industrial experience, experience in RE, and familiarity with the problem domain under study. To evaluate each subject's familiarity with the domain under study we will ask all subjects to take a placement test. This test will ask a few questions about the domain and the results will determine who is familiar with the domain, who is not, and how much. Based on the results, we will divide subjects into four kinds: 1) domain aware: those who score high on the placement test, 2) domain generalist: those who score medium on the placement test, 3) domain analogist: those who score low on the placement test, and 4) domain ignorant: those who score zero or next to zero on the placement test.

In the main setting of the experiment, domain awares and domain ignorants become the main subjects. The domain aware subjects will attend a tutorial session about the domain under study in order to strengthen their domain awareness and thus to sharpen the difference between them and the domain ignorant subjects. The domain ignorant subjects will remain completely uninformed of the domain. Three kind of equal sized teams will be formed using the two kinds of subjects:

1) all-domain aware teams, A teams, each consisting of only subjects from the domain aware kind,
2) all-domain ignorant teams, I teams, each consisting of only subjects from the domain ignorant kind, and
3) mixed teams, M teams, each consisting of a blend of subjects from both kinds and each having a different blend.

In other settings, we are going to use different numbers of ignorants, awares, generalists, and analogists to see the impact of each and of each configuration on the effectiveness of the entire team.

Apart from group membership, the assignment of subjects to the teams is random. The main experiment session will be held separately for each team. All teams will do the same activity with the same CBS and the same artifact for the same amount of time.

In order to be able to interpret and explain the results of the requirements elicitation experiments, we intend to give each subject a creativity test adapted from the one developed by Frank Williams to detect the presence of significant differences in personal creativity that could affect and explain the outcomes of the experiments.

Each team will be given the description of some CBS in the domain under study. Its task is to understand the system as well as it can and then to write as many requirements for the CBS as it can. The requirements will be written in unstructured natural language. Alternatively, each team will be asked to inspect some requirements document for some CBS in the domain under study and to report as many defects as it can. The defects will be described in unstructured natural language. An experimenter functions as an observer, recording the progress of each team. For any M team session, the observer notes which requirements or defects are discovered by which participant. This information will be used in the analysis of the results to correlate each requirement or defect with the type of subject that discovered it.

After the end of a team's session, it will receive a gold version of what it was supposed to generate. Subjects will be then asked to comment on the requirements or defects that they found and on those that they missed from the gold version. These comments will be reviewed by the experimenters to enable them to better match the items found by subjects and those of the original set.

3.2 Observation

In an observation study, we want to observe the subjects directly to investigate the effects of different distributions of knowledge on the effectiveness of the project teams. With any project, we would use various data collection techniques, including interviews, questionnaires, and observations.

We would like also to be able to attend an occasional elicitation and inspection session and to control the distribution of domain aware and domain ignorant people in the session's team and then to measure the effectiveness of the team in its elicitation or inspection.

3.3 Examination of Project Histories

We would like to examine CBS requirements and other artifacts from past, current, and future projects. With a past project, perhaps we could determine after the fact the distribution of knowledge on the project and correlate that distribution with effectiveness measures.

3.4 Survey

A survey of experts is another type of study we are planning to do. We could send out Web-based questionnaires asking practitioners in industry, to see whether they have had any exposure to the effect of domain ignorance in their past RE activities.

3.5 Case Studies

At the time of writing, we do not have any specific plan for conducting case studies. The common limitation of case studies is that their findings are not easily generalized to larger population of subjects and other project situations. However, due to the constraints of conducting a controlled experiment in a real-world project settings, we might end up doing a few case studies at different companies with professionals as subjects to find out whether the same sort of results are obtained within industry as those achieved from controlled experiments.

4 Analysis

We plan to apply both quantitative and qualitative metrics to data about each team's task. First, we will count the number of requirements or defects each team has found. Even though this is a simple measure to obtain, it would not make much sense to evaluate a team's performance based on only the number of requirements or defects it found. Therefore, we plan to consider also the quality of the findings of each team. For the elicitation experiment, we will measure the quality of a generated requirement using the characteristics of a good requirement in the IEEE 830 Standard [12]. This standard says that a good requirement is feasible, valid, unambiguous, verifiable, modifiable, consistent, complete, traceable, and concise. We will have several domain experts evaluate each requirement according to these properties, and then we will correlate the experts' evaluations.

For the inspection experiment, we will have several domain experts categorize each defect, e.g., an ambiguity defect, and then we will correlate the experts' categorizations. The reason to categorize defects is to see whether different factors including domain knowledge influence the type of defects found by each team and each participant. For example, we might learn that inconsistencies among requirements are more likely to be found by domain ignorants.

5 Progress and Future Works

To date, we have conducted two pilot studies. The first was a controlled experiment on requirements elicitation, with four 3-person teams. The all aware team surprisingly produced the least amount of requirements while the all ignorant team generated a rather large number of requirements; but the best result obtained by the team containing 2 domain awares and only one domain ignorant. While the original hypothesis [2] was that the best team to have is all awares except one domain ignorant with the maximum expertise and just enough ignorance to ask questions, the all ignorant team was essentially the same, saying that expertise was not important at all. The second pilot study was a controlled experiment on requirements inspection. The second study had three 2-person teams of all the possible combinations; the results appeared to be almost equal for the mixed and aware team but the ignorant team did poorly in respect to the number of defects detected.

Since these studies were only pilots, their results are not generalizable. However, we have learned how to improve the experiment. The most reliable results though would be obtained from an experiment done in an industrial project setting with a real-world system and professionals as subjects.

References

1. Brooks, F.P.: Mythical Man-Month. Datamation 20, 44-52 (1974)
2. Berry, D.: The importance of ignorance in requirements engineering. Journal of Systems and Software 28, 179-184 (1995)
3. Basili, V.R., Caldiera, G., Rombach, H.D.: The goal question metric approach. Science 2, 1-10 (1994)
4. Berry, D.: The importance of ignorance in requirements engineering: An earlier sighting and a revisitation. Journal of Systems and Software 60, 83-85 (2002)
5. Randell, B., Buxton, J.N.: Software Engineering Techniques: Report of a conference sponsored by the NATO Science Committee. Brussels, Scientific Affairs Division, NATO (1969)
6. Dieste, O., Juristo, N., Shull, F.: Understanding the Customer: What Do We Know about Requirements Elicitation? IEEE Software (2008)
7. Kenzi, K., Soffer, P., Hadar, I.: The Role of Domain Knowledge in Requirements Elicitation: An Exploratory Study. In: Conference The Role of Domain Knowledge in Requirements Elicitation: An Exploratory Study. (2010)
8. Ferrari, R., Madhavji, N.H.: The Impact of Requirements Knowledge and Experience on Software Architecting: An Empirical Study. WICSA (2007)
9. Carver, J.C., Nagappan, N., Page, A.: The Impact of Educational Background on the Effectiveness of Requirements Inspections: An Empirical Study. IEEE Transactions on Software Engineering 34 (2008)
10. Rubens, J.: Business analysis and requirements engineering: the same, only different? Requirements Engineering 12, 121-123 (2007)
11. Barker, M.: BA Careers - Domain knowledge versus analysis skills. vol. December 20 (2010)
12. Berenbach, B., Paulish, D., Kazmeier, J.: Software and Systems Requirements Engineering: In Practice. (2009)

Managing Imperfection in Requirements: a Method and a Jigsaw Puzzle Metaphor

Maria Pinto-Albuquerque[1,2], Awais Rashid[1],

[1]School of Computing and Communications, Lancaster University, United Kingdom
[2]Instituto Universitário de Lisboa (ISCTE-IUL), Lisboa, Portugal
maria.albuquerque@iscte.pt, marash@comp.lancs.ac.uk

Abstract. Effective requirements engineering in the presence of imperfection remains a major research problem and there is a lack of metaphors to aid communication during consultation with stakeholders. We propose a metaphor to promote and improve communication between stakeholders and requirements engineers when working together on requirements documentation, to detect and handle imperfection. The metaphor is based on jigsaw puzzles, where each puzzle piece represents a concern (i.e., a set of requirements). When the requirement's text contains imperfections potentially leading to conflicts with other concerns, the respective puzzle pieces have a matching that almost fits but not perfectly. We argue that using such a metaphor fosters team work and communication towards detecting and analyzing imperfections in stakeholder consultation meetings. It also makes stakeholders feel that imperfections in requirements documentation are their problem too. Having the jigsaw puzzle, a game widely known and very easy to learn and play, provides a good metaphor for group work among persons with heterogeneous background, and introduces fun in a work usually perceived as boring. Together with the jigsaw puzzle metaphor we present a method that receives a text document and through the integration of tools and heuristics supports the detection of imperfections, in particular conflicts. The requirements engineer can then select the most pertinent conflicts and require the appropriate tool to produce a jigsaw puzzle to be used in a consultation meeting with stakeholders. We have already conducted an experiment, which indicated interesting results, in particular an effective improvement of communication. We plan to perform two more experiments.

Keywords: Requirements, Communication, Team work, Stakeholders, Imperfection, Conflict Detection, Jigsaw puzzle metaphor, Empirical evaluation

1 Problem and its Relevance

The development of software is a complex task, requiring the acquisition and processing of a huge amount of information. Software engineering involves a large number of decisions to be taken at the right time. These decisions are based on information and produce information that should be recorded and presented in an accessible manner for both the software engineers and the stakeholders.

Providing appropriate and usable mechanisms to represent information is difficult because information in software development is heterogeneous, inputted and used by a heterogeneous professional community and also inherently not perfect. This is particularly true in requirements engineering. Existing software development approaches aim to come to a perfect set of requirements. But they also acknowledge the difficulty of defining requirement specifications with desired quality. Due to this difficulty some approaches have started to include tools, like iteration and heuristics, to ensure the quality of the resulting software system. But for a non-trivial system, it is unrealistic to assume that the development is performed in an ideal case where all the imperfect information will become perfect, the requirements will not change and/or it will be possible to correct all the problems caused by wrong assumptions. Furthermore, it is unrealistic that it will be possible to perform the large number of iterations needed as these are restricted by time and financial constraints [13].

Requirements are pieces of information where imperfection, like incompleteness, ambiguity, and conflict are inherently present. It is not realistic to think that imperfection can be removed before or during systems engineering. Thus, decisions during development are almost always made in the presence of imperfection. The best we can aim is at making explicit the information on the imperfection facets of requirements and deal effectively with such imperfection, for instance, resolve it where possible or be mindful of it when making decisions [12].

By imperfection we mean things like incompleteness, misplacement, ambiguity, and conflict. In our research we focus on conflicting or ambiguous requirements expressions, and not conflict due to different interests of stakeholders and/or software engineers.

The ultimate reason to address imperfection in requirements is that if imperfection is not made explicit and appropriately handled, it will be supposed that requirements are perfect. Development will proceed and the system produced will contain errors or unwanted characteristics. Big disasters, costing even human lives, due to imperfection in requirements are well known in systems engineering history (e.g., Therac-25 [11], Mars Climate Orbiter [18], etc.). It is also known that the sooner the errors are corrected the less costly they are [5].

2 Proposed Solution

2.1 Novel Aspects of the Work

We propose a metaphor based on jigsaw puzzles in order to promote and improve communication amongst system stakeholders and between stakeholders and requirements engineers. This metaphor is to be used during consultation meetings with stakeholders, fostering group work to review requirements documentation in order to detect and handle imperfection. In these meetings, typically requirements engineers want to focus in a small number of concerns (say 4 to 6) [17] and facilitate involvement of stakeholders. Our hypothesis is that this jigsaw puzzle with imperfect interlocking shapes between pieces (representing concerns with potential conflict) fosters team work and communication towards detecting and analyzing imperfections in requirements meetings. When the requirement's text contains imperfections potentially leading to conflicts with other concerns, the respective puzzle pieces have a matching that almost fits but not perfectly. We also foresee that this metaphor will improve stakeholder's awareness that imperfections in requirements are their problem too and thus increase their commitment to cooperate in solving those imperfections. We believe this will happen because both stakeholder's and engineers have to work with the same common "document", i.e. the puzzle, and not each with its own copy of requirements text.

We propose a method to support the management of imperfect information in requirements. This method receives requirements documents written in natural language, and integrates existing tools alongside with heuristics to detect, classify, and prioritize imperfections. For the most pertinent cases of requirements conflicting, the requirements analyst(s) has the option to produce, supported by a developed tool, a set of jigsaw puzzle pieces to be used in consultation meetings with stakeholders to review that part of the requirements documentation. The method prescribes that the remaining imperfections are handled through annotations. These annotations record information about the imperfection and describe how to proceed in order to manage it (e.g., what to ask, to whom). The method is to be iteratively repeated, as long as the requirements team considers it useful and according to time and budget constraints. This means that, at different moments during development, diverse parts of the requirements can be selected to be discussed in consultation meetings with stakeholders, or amongst analysts. For each set of requirements a different jigsaw puzzle set is produced.

We propose a solution based on a visual metaphor because we believe metaphors may provide a good solution to address the needs of making imperfections in requirements explicit via visualization. A visual metaphor is an analogy that underlies a graphical representation of an abstract entity or concept with the goal of transferring properties from the domain of the graphical representation to that of the abstract entity or concept [4, 9]. The more complex visual metaphors (more complex than two- or three-dimension geometric ones) have been applied in its vast majority to artifacts and software that already exist (to show metrics, for program comprehension, for reverse engineering) [4]. Using visualization to support creation and decision in software development requires a different mindset: to provide visualizations for artifacts and organization that are not known, they are being built. We believe a well-assembled metaphor, making an analogy with a building process where in the beginning we do not know the organization but it starts to appear, can provide interesting tools for software development, focused on its "being built" nature. The jigsaw puzzle game provides a good metaphor for building something from different pieces that have to be correctly assembled to yield a final common product.

In order to accommodate usage by different professional profiles that cooperate in software development we should provide a metaphor that builds on a well-known concept to be easily used by the broadest professional profiles. The jigsaw puzzle, a game widely known and very easy to learn and play, potentially provides a good metaphor for group work among persons with heterogeneous background. Last but not the least we should not forget that these consultation meetings with stakeholder's are usually boring and being the jigsaw puzzle a game introduces fun in work!

2.2 Research Methodology

We planned to perform three experiments. These experiments are concerned with understanding how requirements engineers and system stakeholders communicate and work together to detect, and analyze imperfections present in requirements documentation for a system. In particular, we want to study how requirement engineers and stakeholders communicate and handle imperfection with the jigsaw puzzle metaphor we have developed. Our hypothesis is that this jigsaw puzzle metaphor with imperfect interlocking shapes between pieces (representing concerns with potential conflict) fosters team work and communication towards detecting and analyzing imperfections in requirements and improves commitment of stakeholders in resolution of imperfection.

In the experiments we are going to emulate the "real-life context" of a consultation meeting between requirements engineers and stakeholders. In these meetings with stakeholders the investigators will perform the role of requirements engineers and facilitators, while participants will play the role of stakeholders.

These experiments have a mixed philosophical stance: positivist and constructivist. Thus the experiments will be confirmatory once they will be used to test the hypothesis described above. The experiments will be exploratory once they will be used to understand the capabilities and problems of the proposed metaphor, eventually leading to new hypothesis. When comparing the empirical results with the proposed theory, the conclusions may indicate that the results confirm the theory (as we expect), i.e. the proposed jigsaw puzzle metaphor facilitates and improves communication and work in order to handle imperfection in meetings with stakeholders. It may happen that the results contradict the theory. In this case we would have to reformulate our propositions. As our experiments have also an exploratory nature, the development of a rich case description will be very useful once it will enable to understand the capabilities and problems of the proposed metaphor, which can lead to an improvement of our theory.

We already performed an experiment with results indicating an effective improvement of communication during consultation with stakeholders. We presented a jigsaw puzzle set composed of 4 cardboard pieces with size 10 cm x 10 cm, and representing 4 concerns of the Crisis Management System [7]. Each piece has bumps and dips in one or more edges, which are a cue to how they should be assembled to build the puzzle. The participants were asked to try to build the puzzle. When doing this they discovered that there is a way the pieces fit but not perfectly. At this moment they were asked to read the text in the pieces, and scan what could be the possible sources of imperfection. The text written in a piece representing a concern is the same as in the requirements documentation. To improve readability we cut some not necessary text, made some abbreviations, displayed the text in list mode, and used upper case letters to stress the "topic" of each requirement. Figure 1 shows a picture of the cardboard puzzle after being used in the experiment and Figure 2 shows the text for Availability and Reliability as it was published [7]. In this experiment participants discovered all the imperfections we were aware and some others we had not thought of a priori. Just to give an example, when looking at Availability and Reliability texts, triggered by the non-fitting interlocking shapes, participants could guess that requiring "max. failure rate: 0,001%" as in Reliability would not be compatible with allowing "downtime for maintenance: 2 h each 30 d" as Availability allows. In fact, if one discounts 24h to the hours a year contains, it is achieved a downtime of 0,274%, higher than 0,001%. However, if we observe from a different angle we can see that the insufficiency may instead inhabit in knowing if "downtime for maintenance" (term used in the Availability requirement) is to be interpreted or not in the same way as "failure" (term used in the Reliability requirement). The participants in the experiment also pointed out this other possible imperfection.

Some of the behaviors and emotions we perceived are now described. When trying to build the puzzle, the participants started to propose to others what tactic to use, like: "make first the corners" and we could perceive they were exploring how to work in group. After this initial phase, participants collaborated as a group, not having problems in posing their questions or making comments, and saying things like: "wait a minute…" and then explaining their reasoning and offering their comments. The participants were handling the pieces, even taking them up of the table, and showing them to others. They used the direction of the text written on the pieces as another visual cue (in addition to the interlocking shapes) that helped to understand how the pieces should go together. During the phase of scanning for imperfections, the participants kept working as a group, and when faced with a possible imperfection, participants discussed among them. After achieving a consensus, one participant hand-wrote the group conclusion on the piece using numbers and/or letters to refer to the different pieces and phrases inside a piece. In some imperfection cases the participants proposed a common remedy for the imperfection. Some of the suggestions for improvements that come out from this experiment are: to have the possibility to identify, univocally, each requirement (each phrase in a piece) so that it can be referred to in comments anywhere in the puzzle; make the pieces bigger so that there is more free space for hand-written comments; and reinforce the jigsaw puzzle metaphor common cues such as the use of color on the surface of the puzzle pieces. We also discussed with participants the question: if it would be preferable to have virtual puzzle pieces instead of the presented cardboard pieces. Participants unanimously preferred the physical cardboard pieces, because of the possibility to handle the pieces, picking them up and showing to others while discussing, and also the possibility of writing on the pieces. But the idea of having (also) the virtual pieces supported, for instance by a digital imaging table is an appealing one. These digital supported pieces could provide instant visual feedback on decisions participants make when facing an imperfection (like changing a requirement text, or adding/deleting a requirement or concern). The exploration of the digitally supported puzzle pieces provides interesting questions for future work. Some of these questions are: how will the virtual puzzle adjust according to decisions participants make, like delete a concern and thus a puzzle piece? What are the kinds of decisions allowed? How can the aspects participants appreciated in the cardboard pieces be maintained and adapted to the digital media?

Fig.1. The cardboard jigsaw puzzle after being used in the first experiment.

• **Availability**
– The system shall be in operation 24 hours a day, everyday, without break, throughout the year except for a maximum downtime of 2 hours every 30 days for maintenance.
– The system shall recover in a maximum of 30 seconds upon failure.
– Maintenance shall be postponed or interrupted if a crisis is imminent without affecting the systems capabilities.
• **Reliability**
– The system shall not exceed a maximum failure rate of 0.001%.
– The mobile units shall be able to communicate with other units on the crisis site and the control centre regardless of location, terrain and weather conditions.

Fig.2. The text for Availability and Reliability as it is published [7].

We are now preparing a second experiment to explore pieces surface and eventually take profit from the jigsaw puzzle common cue of having an image in background guiding how the pieces should be assembled. The pieces size will be bigger: 12 cm x 12 cm, and the requirements will be labeled for reference in discussion and report. We will perform 3 sessions with 3 different groups using 3 systems, and each group will work as control group for a group in another session. For instance, in the 1st session the group will work with system A with jigsaw puzzle metaphor, and with system B with plain text. In the 2nd session, the group will work with system B with jigsaw puzzle metaphor, and system C with plain text. The 3rd session will work with system C with jigsaw puzzle metaphor, and system A with plain text. With this second experiment we want to confirm our hypothesis further and explore the usage of piece surface that come out as a suggestion from the first experiment. We are, also developing a software tool to generate the jigsaw puzzle pieces.

3 Related Work and Conclusion

There are a number of research areas that have focused on particular aspects of imperfection in information such as conflict management or evolution.

There are also, some methods that provide decision support for specific software activities that are hampered by imperfect information. They extend the expressive capabilities of the development process, through adding models, describing important properties of imperfect information, for instance by means of probability theory or fuzzy logic [21, 10, 15, 13]. There has been an effort to support imperfect information in software development tools, across the life cycle. This is especially true in the work of Noppen [13]. This work focuses on setting up the foundations for the support of imperfect information in software development. Communication and user interaction issues are not the focus of this work, recognizing that the user interaction with the imperfection models (which are mathematical models) can be difficult, in particular since the intended users do not necessarily know them. In fact, it is necessary to study what kinds of imperfect information exist and which ones are relevant for effective decision support and how. Skeels et al work gives a step in this direction in what concerns the uncertainty type of imperfection [16]. Our work aims to explore the potential synergy between Information Visualization and SE Visualization (as envisioned by Gotel et al [6]) to provide good metaphors for the integration and support of imperfect information in software development tools. In fact, visual metaphors have, from a long time, been used to represent information in software engineering. The diagrams are geometric-based metaphors and the tree (even the mathematical concept) is a metaphor. One example of more sophisticated visual metaphors is the city metaphor, which was used to visualize software code [3, 8, 14, 19, 20]. In the city metaphor a building or a district represents an object-oriented class and the visual characteristics are used to depict software characteristics and metrics. Another interesting metaphor is the landscape metaphor where landscapes are used to represent software systems [1]. Boccuzzo [2] propose the usage of the concept of well-shaped graphical visualization to represent that the corresponding artifact is well designed. This shows that the metaphor "language" can be used to express information about the artifacts that are being represented.

We believe the presented jigsaw puzzle metaphor promotes and improves, in a novel way, communication between stakeholders and requirements engineers when working together on requirements, to detect and handle imperfection.

References

1. Balzer, M., Noack, A., Deussen, O., Lewerentz, C.: Software landscapes: Visualizing the Structure of Large Software Systems. In: VisSym 2004, Symposium on Visualization, pp. 261—266. Eurographics Association (2004)
2. Boccuzzo, S., Gall, H.C.: Cocoviz: Towards cognitive software visualization. In: IEEE Int. Workshop on Visualizing Software for Understanding and Analysis. IEEE CS(2007)
3. Charters, S.M., Knight, C., Thomas, N., Munro, M.:In: Visualisation for informed decision making; from code to components. In: International Conference on Software Engineering and Knowledge Engineering (SEKE '02), pp. 765—772. ACM Press (2002)
4. Diehl, S.: Software Visualization. Springer (2007)
5. Fagan, M.E.: Design and code inspections and process control in the development of programs. IBM Systems Journal 15(3), 182—211 (1976)
6. Gotel, O., Marchese, F., Morris, S.: The Potential Synergy between Inf. Visualization and SE Visualization. In: 12th Int. Conf. Inf. Visualization, pp. 547—552. IEEE CS (2008)
7. Kienzle, J., Guelfi, N., Mustafiz, S.: Crisis Management Systems: A Case Study for Aspect-Oriented Modeling, version 1.0.1, http://www.cs.mcgill.ca/~joerg/taosd/TAOSD/TAOSD_files/AOM_Case_Study.pdf. May (2009)
8. Knight, C., Munro, M.C.: Virtual but visible software. In: International Conference on Information Visualisation, pp. 198—205. IEEE Computer Society (2000)
9. Lakoff, G., Johnson, M.: Metaphors We Live By. The University of Chicago Press (1980)
10. Lee, J., Kuo, J., Hsueh, N., Fanjiang, Y.: Trade-off requirement engineering, In: Lee, J.(ed.) Software Engineering with Computational Intelligence, pp. 51—72. Springer (2003)
11. Leveson, N.: Medical Devices: The Therac-25. In: Safeware: System Safety and Computers. Addison-Wesley (1995)
12. National Science Foundation: NSF Workshop on the Science of Design: Software and Software Intensive Systems. In: Sullivan K. (ed.), http://www.cs.virginia.edu/~sullivan/sdsis/SDSIS%20Preliminary%20Report%20020210.pdf. (2003)
13. Noppen, J.: Imperfect Information in Software Design Processes. Ph.D. Thesis, Enschede, The Netherlands (2007)
14. Panas, T., Berrigan, R., Grundy, J.: A 3d metaphor for software production visualization. In: International Conference on Information Visualization, p. 314. IEEE CS (2003)
15. Shaw, M.: Truth vs Knowledge: The Difference Between What a Component Does and What We Know It Does. In: 8th Int. Work. on Software Specification and Design (1996)
16. Skeels, M., Lee, B., Smith, G., Robertson, G.: Revealing Uncertainty for Information Visualization. In: AVI, Int. Work. Conf. on Adv. Visual Interfaces. ACM Press (2008)
17. Sommerville, I, Sawyer, P.: Requirements Engineering: A Good Practice Guide. Wiley (1997)
18. Stephenson, A. G., LaPiana, L. S., et al.: Mars Climate Orbiter Mishap Investigation Board Phase I Report, NASA (1999)
19. Wettel, R., Lanza, M.: Program Comprehension Through Software Habitability. In: 15th International Conference on Program Comprehension, pp. 231—240. IEEE CS (2007)
20. Wettel, R., Lanza, M.: Visualizing Software Systems as Cities. In: 4th IEEE Int. Work. on Visualizing Soft. For Understanding and Analysis, pp. 92--99. IEEE CS(2007)
21. Yen, J., Lee, J.: Fuzzy Logic as a Basis for Specifying Imprecise Requirements. In: 2nd IEEE Int. Conference on Fuzzy Systems (FUZZ-IEEE'93), pp. 745—749. IEEE CS (1993)

Evolutionary Process Improvement Using Method Increments

Kevin Vlaanderen
(Promotor: Prof. dr. Sjaak Brinkkemper, Co-promotor: dr. Inge van de Weerd)

Utrecht University,
Department of Computer Science,
P.O. Box 80.007,
3508 TA, Utrecht, the Netherlands,
k.vlaanderen@cs.uu.nl

Abstract. Although a lot of research is produced in the fields of requirements engineering and software product management, insufficient knowledge seems to be transferred between researchers and professionals, and among professionals themselves. Furthermore, the solutions that are produced by the scientific community are often too big or too complex to implement as a whole. In this dissertation research, we will focus on incremental process improvement by elaborating on the concept of method increments. The envisioned result is an incremental approach for process assessment, process improvement, and process execution, based on method engineering techniques and tools. The approach is implemented in the Online Method Engine; a holistic solution that improves knowledge dissemination and effective process improvement for SPM.

1 Introduction

In the area of software engineering, many research has been performed related to process improvement, many of which published during the European Workshop on Software Process Technology and the International Conference on Product-Focused Software Engineering. Much of the research performed in this field resulted in frameworks to assess processes, and guide and support process improvement efforts, such as CMM [11], CMMI [10] and SPICE [6].

In many cases, these approaches were prescriptive in nature, in the sense that they described an optimal situation that would work best for all companies, despite their differences. During the early 90's, a small group of researchers decided to split off from this line of research. Initially, this resulted in a field of research named Method Engineering, with a focus on creating a research framework for methods and tools for information systems development [4]. This quickly evolved into the area of Situational Method Engineering, representing the belief that no one method is applicable in all situations, and that each process improvement requires a characterization of the current situation, resulting in the construction of a customized or situational method [8].

To bridge the gap with the original process improvement efforts, research during the last five years has mainly focused on supporting evolutionary process improvement through incremental method evolution [17]. During an explorative case study, van de Weerd et al. [17] defined the atomic types of adaptations that can be made to an existing method, called *method increments*. Insight into these method increments enables focused improvement of software development processes based on localized issues and improvement possibilities. In theory, it allows the linkage of locally identified problems to specific changes to the current method.

In line with this research, we now need to deepen our understanding of these method increments, by determining how they appear in practice, and how they are related to research and concepts regarding process assessment, situational method engineering, and process improvement. Therefore, the main research question of this research is as follows:

> How can method increments be employed to support incremental method evolution?

Such a question not only entails research into the workings of method increments, but also insight into the social, cultural, and financial implications of adaptations to existing processes.

1.1 Domain

Although we envision that the research resulting from this project is generally applicable to the field of software development, we have chosen to work within the specific subdomain of software product management (SPM). SPM is a term that has come to existence due to the shift that many software companies are currently experiencing, from project-driven software development to market-driven product development. With this change, the range of influences, issues and benefits that these companies need to deal with change tremendously. Due to the relative young age of this subdomain, software product management could profit significantly from the ideas that we propose.

During the past two decades, the field is seeing a steady flow of new techniques and method, developed both by academia as well as practitioners. The main share of these publications consists of conceptual or practical solutions to problems within the field (followed by theoretical elaborations and state-of-the-art papers). These solutions are either found within industry and abstracted into scientific terms, or developed within the academy and validated in practice. The problems that are dealt with can vary from distributed prioritization of requirements to the assessment of product management processes.

Unfortunately, several problems inhibit effective SPM. Firstly, hardly any schooling is available for product managers. This is related to the fact that product managers have often evolved into that function after first performing a development- or project management function. Because of this, a lot of product managers miss essential skills and knowledge that are required for performing their function effectively. In addition, the transfer of knowledge between academy and industry is far from optimal, reducing much of the fundamental research currently produced to a state where it is not helpful anymore, throwing away a lot of potential. This last issue has been recognized more than once, amongst other during the keynote presentations of the recent RE and RefsQ conferences, and also by Gorschek et al. [7] in their paper on technology transfer in practice.

2 Proposed Solution

Although the concept 'method increment' is not new [17, 2], our understanding of its precise meaning is not profound enough to effectively employ it during process improvement efforts. The main contribution of this research lies in the definition of method increments in terms of size, contents, and above all their applicability for situational, incremental method engineering.

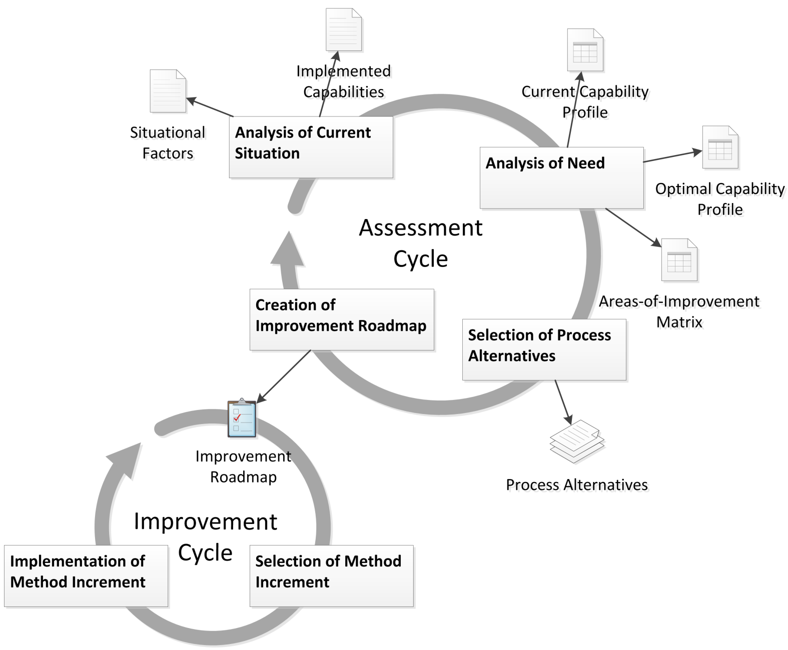


Fig. 1. Incremental Process Improvement

Based on earlier work by Bekkers et al. [2] and influenced by process improvement approaches such as the Deming cycle of Plan, Do, Check and Act [5] and the Quality Improvement Paradigm (or experience factory) [1], a high-level approach to incremental method evolution has been proposed in Vlaanderen et al. [15](cf. Figure 1). The approach consists of one major process improvement cycle with one subcycle. The starting point for each process improvement is an analysis of the current process, based on which a maturity profile can be calculated. The situational factors [3] of the company are used to determine an optimal maturity profile. By calculating the delta between these two, the required process improvement is determined. This process improvement is further detailed by relating it to suitable method fragments that can be combined into a new process that improves the company's process.

This approach leaves room for a lot of interpretation, and can be executed using many different approaches. During this research, we will look at ways in which method increments can be successfully incorporated into the assessment and improvement cycles (with an emphasis on the latter). An important step in this process is determining how method increments appear in the field, if they appear at all. Based on this research, we should determine how method fragments relate to the concept of method increment, which boils down to the question whether method fragments can be regarded as puzzle pieces. Another aspect of this research is the question of translating process assessment results into meaningful method adaptations or method increments. And to conclude, it is very important to consider the social and cultural implications of such changes, requiring a considerable part of this research to be dedicated to the problem of research communication and advice.

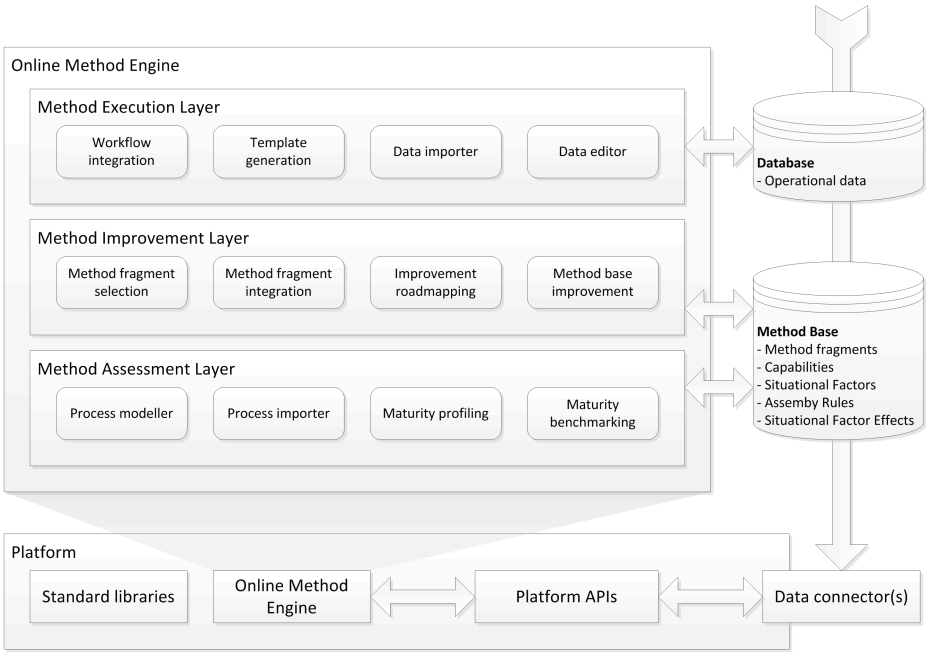


Fig. 2. Online Method Engine

To further support the field of SPM, results of the research performed in this project will constantly be applied in a knowledge infrastructure with the aim of disseminating SPM knowledge and enabling effective process improvement. A first move towards this goal has been made by elaborating a conceptual description of such a knowledge infrastructure, called the Online Method Engine (OME) [15].

An architectural vision on the OME is depicted in Figure 2. At the bottom, the development platform is shown, including the programming language, framework, libraries, etc. On top of the platform, the OME is shown with three functional layers. Each layer contains several functional components, shown by the rounded boxes. The three layers are:

Method Assessment This layer deals with determining the current maturity level within a functional domain.

Method Improvement This layer deals with improving (parts of) a method by selecting improvement actions, determining an improvement roadmap, and applying improvement actions to a method.

Method Execution This layer supports the implementation and execution of an improved method by providing workflow support and on-demand templates.

On the right hand of the figure, two databases are shown; one for the method engineering related data such as method fragments and situational factors, and one for the method execution related data such as requirements and planned releases. These database are connected to the system using separate data-connectors, which allow connection to any suitable database, ranging from a MySQL database to the database of a third-party requirements engineering tool.

3 Novelty

The proposed solution relies heavily on method engineering techniques. The field of method engineering has seen an increasing amount of interesting approaches and techniques over the last twenty years [14, 13, 8, 4]. The coverage of these techniques ranges from the modeling of processes and systems to the situational construction of new ones.

Some method bases have actually been implemented, such as OPF [9] and the CREWS method base [12]. However, for practitioners (the actual method users), retrieving these method fragments and using those in their daily work can be cumbersome. A prerequisite is that the method user should be aware of the exact method fragment that he or she is searching for. In addition, the method user must know what to do with the retrieved method fragment, how to interpret it, and how to implement it in the organization. The current method bases and knowledge infrastructures are too hard to use for many practitioners. They do not always know exactly what they are looking for, nor how to implement a formal method description in the processes of their organization.

To overcome these problems, this solution comprises an incremental approach for process assessment, process improvement, and process execution, resulting in a holistic solution based on method engineering techniques and tools. The Online Method Engine proposed here can not only be used to store and retrieve method fragments, but also to assess an organization's current processes, create an advice based on this assessment, and implement this advice in the organizations processes and tools. Furthermore, existing method bases are extended by not only providing a repository in which method fragments are stored, but also the opportunity for users to assess their own processes and investigate which ones should be improved, thereby improving knowledge dissemination. Furthermore, the company's tooling infrastructure can be directly aligned with the method improvement by automatically configuring templates and work-documents.

4 Research Approach

It should be no surprise that the idea stated here is very complex. In addition to a technical challenge, it has a major soft component, as we are dealing with processes that are performed by human beings, and that per definition require a lot of creativity. Introducing changes to these processes will result in resistance and reluctance, and without adequate attention to the soft side of process improvement, any improvement projects using the situational assessment method and the Online Method Engine will surely fail.

Several subquestions need to be addressed in order to answer the main research question. One of the topics that needs to be covered is the relation between method fragments, assessment results, and method increments. What are the puzzle pieces that we work with during an improvement cycle? How can method knowledge, either from academy or industry, be attached to problem statements or wishes?

Furthermore, it is important to determine the optimal size of a method increment. How many changes can be introduced in an existing process before the effects become unmanageable or negatively influenced? Strongly related to this is the roadmap of method increments, which could be a tool to split up big adjustments into smaller, logical pieces that are organized temporally, while also taking into account any dependencies.

Then there is also the softer issue of communication of method increments. Guidance during this step, which could be seen as the 'Do' phase of the Deming cycle, is definitely required, but how is still unclear.

During the research project, it is extremely important to constantly apply research findings to the ongoing development of the OME. First of all, by directly applying research findings to the system, we can constantly validate the proposed solutions. This ensures that the gap between research and practice decreases, instead of actually increasing. Similarly, it helps in keeping the research tuned to the actual problems and demands in industry. The OME can be regularly shown to industry professionals, which leads to important feedback. This feedback can then be directly fed back into the research.

Furthermore, the ultimate goal of this project is to develop a system that helps industry professionals in increasing the maturity of their SPM processes. Although a complete system will not be available for a couple of years, early results can already be used to help achieve this aim.

Unfortunately, the conceptual model of the OME that is presented in [15] has not been fully validated yet. As no concrete system exists at this moment, doing so would have involved asking potential users to imagine themselves using such a system. This is a tremendous effort, especially due to the complexity of it, and would likely not have resulted in a valid response. However, as development continues, the user should not be forgotten. Instead, at several points in time, his or her opinion should be asked and corrections should be made accordingly.

5 Progress

Although several important issues have been already been elaborated, the envisioned system is far from being operational. By zooming in on the earlier developed vision of the PSKI [16], we have gained a better understanding of the requirements. However, since the research is in a very early stage, the system remains mainly conceptual. From this point onwards, each of the areas of the OME needs to be addressed in detail, putting together the puzzle piece by piece.

An important fact that should never be forgotten during the elaboration is that the purpose of the OME is the improvement of SPM processes. As a consequence, we are always dealing with people that bring habits, experiences, and opinions. Overlooking this would result in a system that is too rigid, forcing people into ways of working that they will not accept, thereby foregoing the purpose of the system. However, if it is done right, than the OME has great potential value. We believe that this solution can increase the maturity of the software industry significantly by providing requirements managers, product managers, CTO's and the like with the right tools to optimize their SPM process.

A major challenge that is faced is of a more fundamental academic nature. By developing the OME, we state that processes can be modeled, altered, expanded and downsized as if they were puzzles. Both the computational as well as the social aspect of this statement will prove very challenging. In Vlaanderen et al. [15], these topics have hardly been touched upon. However, they will become very clear once work on the specific parts will begin, and it is important to balance both of them.

References

1. Basili, V.R.: The Experience Factory and its relationship to other Improvement Paradigms. In: Proceedings of the European Software Engineering Conference. pp. 68–83 (1993)
2. Bekkers, W., Spruit, M., van de Weerd, I., van Vliet, R., Mahieu, A.: A situational assessment method for software product management. In: Proceedings of the European Conference on Information Systems (2010)

3. Bekkers, W., van de Weerd, I., Brinkkemper, S., Mahieu, A.: The Influence of Situational Factors in Software Product Management: An Empirical Study. In: IWSPM '08: Proceedings of the 2008 Second International Workshop on Software Product Management. pp. 41–48. IEEE Computer Society, Washington, DC, USA (2008)
4. Brinkkemper, S.: Method engineering: engineering of information systems development methods and tools. Information and Software Technology 38(4), 275–280 (1996)
5. Deming, W.E.: Out of the Crisis. MIT Press (2000)
6. Dorling, A.: SPICE: Software process improvement and capability determination. Software Quality Journal 2(4), 209–224 (1993)
7. Gorschek, T., Garre, P., Larsson, S.B.M., Wohlin, C.: A Model for Technology Transfer in Practice. IEEE Software 23(6), 88–95 (Nov 2006)
8. Harmsen, F.: Situational Method Engineering. Ph.D. thesis, Universiteit Twente (1997)
9. Henderson-Sellers, B.: Process metamodelling and process construction: examples using the OPEN Process Framework (OPF). Annals of Software Engineering 14(1), 341–362 (2002)
10. Höggerl, M., Sehorz, B.: An Introduction to CMMI and its Assessment Procedure (2006)
11. Paulk, M., Curtis, B.: The capability maturity model for software. IEEE Software 10(4), 18–27 (1993)
12. Ralyté, J.: Reusing scenario based approaches in requirement engineering methods: CREWS method base. In: International Workshop on Database and Expert Systems Applications (DEXA). pp. 305–309. No. Dexa 1999, Ieee (1999)
13. Ralyté, J., Jeusfeld, M., Backlund, P., Kuhn, H., Arni-Bloch, N.: A knowledge-based approach to manage information systems interoperability. Information Systems 33(7-8), 754–784 (Nov 2008)
14. Saeki, M.: Object-oriented meta modelling. Object-Oriented and Entity-Relationship Modeling 1021, 250–259 (1995)
15. Vlaanderen, K., van de Weerd, I., Brinkkemper, S.: The Online Method Engine: From Process Assessment to Method Execution. In: To appear in the Proceedings of Method Engineering (ME2011). Paris, France (2011)
16. van de Weerd, I., Brinkkemper, S., Nieuwenhuis, R., Versendaal, J., Bijlsma, L.: On the Creation of a Reference Framework for Software Product Management: Validation and Tool Support. In: International Workshop on Software Product Management (IWSPM). pp. 3–12. IEEE (Sep 2006)
17. van de Weerd, I., de Weerd, S., Brinkkemper, S.: Developing a Reference Method for Game Production by Method Comparison. In: Situational Method Engineering: Fundamentals and Experiences. vol. 244, pp. 313–327 (2007)

Previously published ICB - Research Reports

2011

No 43 (February)
Frank, Ulrich: "Outline of a Method for Designing Domain-Specific Modelling Languages"

2010

No 42 (December)
Frank, Ulrich: "Outline of a Method for Designing Domain-Specific Modelling Languages"

No 41 (December)

Adelsberger,Heimo; Drechsler, Andreas (Eds): "Ausgewählte Aspekte des Cloud-Computing aus einer IT-Management-Perspektive – Cloud Governance, Cloud Security und Einsatz von Cloud Computing in jungen Unternehmen"

No 40 (October 2010)
Bürsner, Simone; Dörr, Jörg; Gehlert, Andreas; Herrmann, Andrea; Herzwurm, Georg; Janzen, Dirk; Merten, Thorsten; Pietsch, Wolfram; Schmid, Klaus; Schneider, Kurt; Thurimella, Anil Kumar (Eds): "16th International Working Conference on Requirements Engineering: Foundation for Software Quality. Proceedings of the Workshops CreaRE, PLREQ, RePriCo and RESC"

No 39 (May 2010)
Strecker, 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 Geschäftsprozess-management-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. V. Gruhn Software Engineering	Design of Software Processes, Software Architecture, Usability, Mobile Applications, Component-based and Generative Software Development
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. E. Rukzio Mobile Mensch Computer Interaktion mit Software Services	Novel Interaction Technologies, Personal Projectors, Pervasive User Interfaces, Ubiquitous Computing
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 Management	Industrial Business Processes, Innovation Management, Information Management, Economic Analyses

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