UNIVERSITEIT TWENTE.

MASTER'S THESIS

Empirical evaluation of change impact predictions using a requirements management tool with formal relation types

A quasi-experiment

R.S.A. van Domburg

Enschede, 26 November 2009

Software Engineering Group
Faculty of Electrical Engineering, Mathematics and Computer Science
University of Twente

Final project (239997)

Business Information Technology
School of Management and Governance
University of Twente

graduation committee

dr. ir. K.G. van den Berg	EWI
dr. A.B.J.M. Wijnhoven	MB
dr. I. Kurtev	EWI
A. Goknil, MSc	EWI

Acknowledgements

First, I would like to thank my graduation committee for their invaluable advice and input. I have regarded our regular meetings as very enjoyable and beneficial to the quality of my work.

Second, I would like to thank all participants for their participation in the experiment and Johan Koolwaaij and Martin Wibbels for their expert insight into the WASP requirements specification. Your input has been very important to attain any research results in the first place.

Third, I would like to thank Ton Augustin, Pieter van Rossum, Klaas Sikkel and Theo Thijssen for their expert insight into requirements traceability. Your comments have enabled me to reflect upon this research from a practical perspective.

Last but not least, I would like to thank Kim Scholte van Mast for her review of my final draft. Your comments have improved the correctness and readability of this thesis.

Abstract

Background: This research was part of a master's thesis to evaluate the impact of using TRIC, a software tool with formal requirements relationship types, on the quality of change impact prediction in software.

Objective: To analyze the real-world impact of using a software tool with formal requirements relationship types; for the purpose of the evaluation of effectiveness of tools; with respect to the quality of change impact predictions; in the context of software requirements management; from the viewpoint of system maintenance engineers.

Method: This research features a quasi-experiment with 21 master's degree students predicting change impact for five change scenarios on a real-world software requirements specification. The quality of change impact predictions was measured by the *F*-measure and the time in seconds to complete the prediction.

Two formal hypotheses were developed. Null hypothesis 1 stated that the *F*-scores of change impact predictions of system maintenance engineers using TRIC will be equal to or less than those from system maintenance engineers not using TRIC. Null hypothesis 2 stated that the time taken to complete change impact predictions of system maintenance engineers using TRIC will be equal or longer than those from system maintenance engineers not using TRIC. The data were collected by a web application and analyzed using ANOVA and χ^2 statistical analyses.

Results: No significant difference in *F*-scores between TRIC and the other groups was detected. TRIC was found to be significantly slower for four out of five change impact predictions. These inferences were made at α=0,05 with a mean statistical power of 54%.

Limitations: The validity was hampered by a limited availability of usable software requirements specifications, experts from industry and theory regarding the impact of change scenarios on change impact prediction. The results cannot be generalized for other software requirements specifications, change scenarios or groups of participants. The condition to provide a solution validation was therefore not met.

Conclusion: Empirical experiments cannot provide a solution validation to new software tools because there are not enough experts in the new tool. Using TRIC to perform change impact prediction on a software requirements specification of low complexity does not yield better quality predictions but does take a longer time.

Table of Contents

List of abbreviations

AIS	Actual Impact Set
ANCOVA	Analysis of Covariance
ANOVA	Analysis of Variance
EIS	Estimated Impact Set
EWI	Faculty of Electrical Engineering, Mathematics and Computer Science
DIS	Discovered Impact Set
FPIS	False Positive Impact Set
GUI	Graphical User Interface
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
MANOVA	Multivariate Analysis of Variance
MB	School of Management and Governance
MoSCoW	Must have, Should have, Could have, Won't have
NR	Non-Randomized
O	Experimental observation
OMG	Object Management Group
QuadREAD	Quality-Driven Requirements Engineering and Architecture Design
ROC	Receiver Operating Characteristic
Std	Standard
SysML	Systems Modeling Language
TBD	To Be Determined
TRIC	Tool for Requirements Inference and Consistency Checking
UML	Unified Modeling Language
URL	Uniform Resource Locator
WASP	Web Architectures for Services Platforms
X	Experimental treatment

1. Introduction

This master's thesis reports on the evaluation of the impact of using a software tool with formal requirements relationship types on the quality of change impact predictions in software. The tool and formal requirements relationship types were developed as part of a requirements metamodel in a research project called QuadREAD, which will be introduced before describing the problem statement, research objective, context and further document structure.

1.1. The QuadREAD Project

This research is conducted at the laboratory of the Software Engineering Group from March 2009 up to and including November 2009. It takes place within the context of the QuadREAD Project, which is a joint research project of the Software Engineering and Information Systems research groups at the Department of Computer Science in the Faculty of Electrical Engineering, Mathematics and Computer Science at the University of Twente. The QuadREAD Project runs from December 2006 up to and including December 2010.

The context of the QuadREAD Project is the early phases in software development processes: the establishment of user requirements based on analysis of business goals and the application domain and the subsequent architecture design of desired systems. The first phase concerns requirements engineering; the second, architectural design. In practice, it appears that these two phases are poorly integrated [50].

The project aims at a better alignment between analysts and architects. The project elaborates on traceability research and focuses on tracing between user requirements and architectural design decisions [50]. Traceability is defined as the degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another [58].

One depiction of traceability in software development is constructed by combining two specializations of traceability in the context of requirements engineering [61]. First, a distinction can be made between pre-requirements specification traceability (forward to requirements and backwards from requirements) and post-requirements specification traceability (forward from requirements and backwards to requirements) [26]. Second, inter-level and intra-level trace dependencies may be distinguished [3]. See Figure 1.

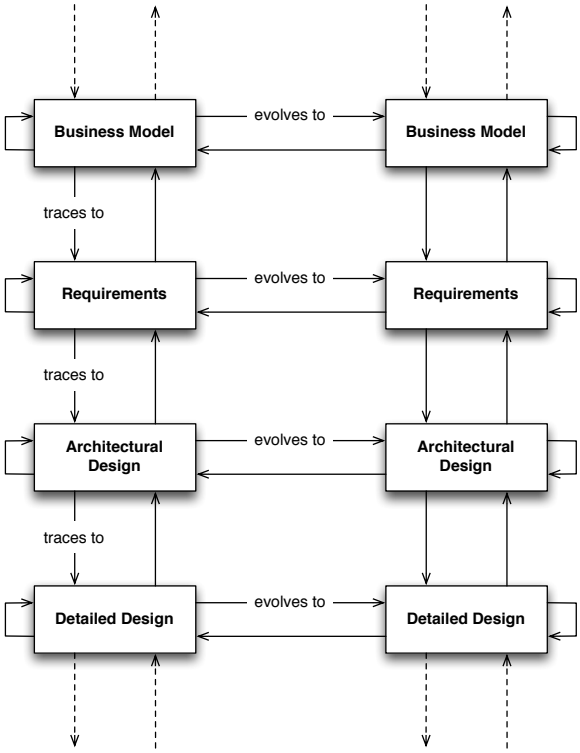


Figure 1: Traceability in software development [61]

Figure 1 shows several types of traceability. For example, requirements elements are traced backwards to elements in business models and forward to elements in the architectural design. Requirements elements may have intra-level dependency relations and may evolve to a new configuration of requirements elements. There are traceability links between artifacts and links representing the evolution or incremental development of these artifacts [61].

In a goal-oriented approach, the QuadREAD Project is developing a framework in which the alignment of requirements engineering and architectural design is supported with practical guidelines and tools. The specific contribution of the project lies in the quantification of quality attributes and tradeoffs in relation to trace information [50].

The project will provide a framework for qualitative and quantitative reasoning about requirements and architectural decisions to ensure selected quality properties. Thereby it will enable decision-making in the quality-driven design of software architectures meeting user requirements and system properties [50].

The research conducted in the QuadREAD Project is intended to have practical applicability by the central role of case studies from participating business partners in the project, includ-

ing Atos Consulting, Chess Information Technology, Getronics PinkRoccade, Logica CMG, Shell Information Technology and Kwards Consultancy [50].

This research is part of the final project of a master's student of Business Information Technology, which is a master's degree program that is headed by the School of Management and Governance of the University of Twente.

The final project is worth 30 European Credits. It is supervised by two assistant professors, one from the School of Management and Governance and one from the Faculty of Electrical Engineering, Mathematics and Computer Science, as well as a postdoctoral scholar and Doctor of Philosophy student from the latter faculty.

Biweekly meetings are held to evaluate the research progress, quality and results. Feedback was also provided by research fellows from the Information Systems Group and business partners participating in the QuadREAD Project, as well as other master's students executing their final project at the Software Engineering Group.

1.2. Requirements metamodel

Research in the QuadREAD Project has contributed a requirements metamodel with formal requirements relationship types to enable reasoning about requirements [25]. Henceforth, this metamodel will be referred to as *the requirements metamodel*. It was constructed based on a review of literature on requirements models. The project also contributed a prototype software tool named TRIC that supports the requirements metamodel. TRIC was illustrated using a single fictional case study featuring a course management system [37].

Based on the case study results, it was concluded that TRIC supports a better understanding of mutual dependencies between requirements, but that this result could not be generalized pending a number of industrial and academic case studies with empirical results [25].

This research on the requirements metamodel can be classified as technique-driven with a lack of solution validation [69]. This classification does not imply that the research quality is poor: papers presenting new technology do not necessarily need to validate the proposed solution, though they should explain why the solution, if validated, would be useful to stakeholders. Validation that a proposed solution actually satisfies the criteria from an analysis of stakeholder goals is a research problem and does not need to be done in a technology paper [70].

1.3. Problem statement

The problem that this research deals with is the lack of solution validation of the requirements metamodel, which can inhibit its adoption because the benefits are not clear. It should be clear to practitioners for which problems a technique has shown to be successful in the real world [69].

1.4. Research objective

The research objective should formulate a means to providing a solution to the research problem. As a starting point, this paragraph compiles a set of solution requirements. A research objective is subsequently formulated.

Solution requirements

The research objective should work towards satisfying two solution requirements:

1. It should evaluate the requirements metamodel as a real-world solution [69] on criteria that were defined in its original research [70].

2. It should be aligned with the goals of the QuadREAD Project, because that is the context in which this research takes place.

The following paragraphs construct a research objective in an iterative fashion by examining these solution requirements more closely.

Evaluation criteria in original research

The original research has defined two evaluation criteria for the requirements metamodel:

1. The number of inconsistent relations in requirements documents
2. The number of inferred new relations in requirements documents

Henceforth, *software requirements specification* is used as a replacement term for requirements documents in the context of software engineering. The term "software requirements specification" is defined in the IEEE Standard Computer Dictionary [58]. It will prove to be useful during upcoming discussions on quality of software requirements specifications, for which the IEEE has well-known recommended practices [59].

Requirements modeling of the case was performed in two iterations using the TRIC software tool, which has support for the formal relationship types from the requirements metamodel.

The first iteration revealed a number of inconsistencies in the software requirements specification. This enabled the researchers to correct these issues. The second iteration reported zero detected inconsistencies [25]. In this case, using formal requirements relationship types led to a higher degree of consistency of the software requirements specification.

In addition to improved consistency, both iterations also reported a greater number of relations than was given initially. The additional relations were inferred by using formal requirements relationship types and led to greater knowledge about the specific requirements in the software requirements specification in the context of requirements modeling [25].

However, the validity of this conclusion may be questioned. Because no tools other than TRIC were used, it could also be concluded that requirements modeling became more effective because any software tool was used. There is no evidence that specifically the formal requirements metamodel that TRIC supports increased the effectiveness of requirements modeling.

Finally, engineers should study real-world problems and try to design and study solutions for them [69]. Likewise, this research should analyze the real-world impact of the formal requirements metamodel by using real-world software requirements specifications and using a real-world impact measure.

Consequently, this research should address this threat to validity by analyzing the real-world impact of the formal requirements metamodel by analyzing TRIC alongside other requirements modeling tools, which support other and less formal requirements metamodels.

Alignment with QuadREAD Project goals

The requirements metamodel contributes to the QuadREAD Project by providing better techniques for change impact analysis, which is necessary for cost-effective software development [6]. It intends to do so by improving the precision of software requirements specifications. Current techniques are imprecise [25] which can reduce the quality of software requirements specifications in terms of ambiguity, modifiability and traceability [59].

Of all users of a software requirements specification, system maintenance engineers are the most concerned with change impact analysis. System maintenance engineers use the requirements to understand the system and the relationships between its parts during requirements management [55].

Indeed, impact is usually associated with maintenance effort [61]. By identifying potential impacts before making a change, system maintenance engineers can greatly reduce the risks of embarking on a costly change because the cost of unexpected problems generally increases with the lateness of their discovery [12]. Having high-quality change impact predictions is thus beneficial to system requirements management.

Goal-Question-Metric approach

Subsequent to the considerations above, a research objective can be formulated according to the goal template of the Goal-Question-Metric approach [73]. The research objective can be formulated as follows;

To improve the adoption of the requirements metamodel and advance the state of the art in change impact analysis, the research should:

> Analyze the real-world impact of using a software tool with formal requirements relationship types; for the purpose of the evaluation of effectiveness of tools; with respect to the quality of change impact predictions; in the context of software requirements management; from the viewpoint of system maintenance engineers.

Operationalizations for this goal are provided in Chapter 2.

1.5. Research method

The research method will involve performing change impact analysis on selected change scenarios on software requirements specifications in two ways: using classic software tools and using the prototype TRIC software tool that supports formal requirements relationship types. Such a research setup involves control over behavioral events during change impact analysis, for which experimental research is the most appropriate [72].

Experimental research has several subtypes, one of them being quasi-experimental research. By definition, quasi-experiments lack random assignment. Assignment to conditions is by means of self-selection or administrator selection [52] such as is the case in our setup with selected change scenarios and a predetermined set of software tools. Consequently, quasi-experimentation is the most appropriate research method.

The quasi-experimental research design is described in Chapter 3.

1.6. Contributions

Through a systematic design and execution of a quasi-experiment to empirically validate the impact of the TRIC software tool on change impact predictions, this research reveals the following:

- Empirical experiments cannot provide a solution validation to new software tools because there are not enough experts in the new tool. This is a phenomenon that will apply to any research regarding new software tools.
- Approximating the experts by training a group of non-experts is difficult to do reliably and hampers internal validity to such a point that an empirical approach to solution validation is infeasible.
- Using TRIC to perform change impact prediction on a software requirements specification of low complexity does not yield better quality predictions but does take a longer time than compared to using Microsoft Excel or IBM Rational RequisitePro.
- It is hypothesized that TRIC is a more intelligent software tool and its benefits will only materialize given a sufficiently complex software requirements specification.
- There is a lack of theory surrounding the nature of change scenarios which poses a reliability issue to any research that deals with them.

1.7. Document structure

This document aims to present the research in a rigorous structure. Such a structure makes it easier to locate relevant information and lowers the risk of missing information [30]. The research is presented as follows:

- **Chapter 2: Background and related work** clarifies how this research relates to existing work, including a description of software requirements specifications, specific requirements, their quality criteria, the requirements metamodel and alternative solutions.
- **Chapter 3: Experimental design** describes the outcome of the experiment planning phase, including goals, hypotheses, parameters, variables, design, participants, objects, instrumentation, data collection procedure, analysis procedure and evaluation of the validity.
- **Chapter 4: Execution** describes each step in the production of the experiment, including the sample, preparation, data collection performed and validity procedure.

- **Chapter 5: Analysis** summarizes the data collected and the treatment of the data, including descriptive statistics, data set reductions and hypothesis testing.
- **Chapter 6: Interpretation** interprets the findings from the analysis including an evaluation of results and implications, limitations of the study, inferences and lessons learned.
- **Chapter 7: Conclusions and future work** presents a summary of the study, including impact, limitations and future work.

A glossary and list of references is presented afterwards.

2. Background and related work

2.1. Introduction

This chapter describes the related work that is relevant for this research. The related areas follow from the research objective, which is repeated here:

> Analyze the real-world impact of using a software tool with formal requirements relationship types for the purpose of the evaluation of the effectiveness of tools with respect to the quality of change impact predictions in the context of software requirements management from the viewpoint of system maintenance engineers.

A conceptual framework for background and relevant work can be developed by relating the keywords in this research objective. The nature of the relationships is discussed in the following paragraphs. See Figure 2.

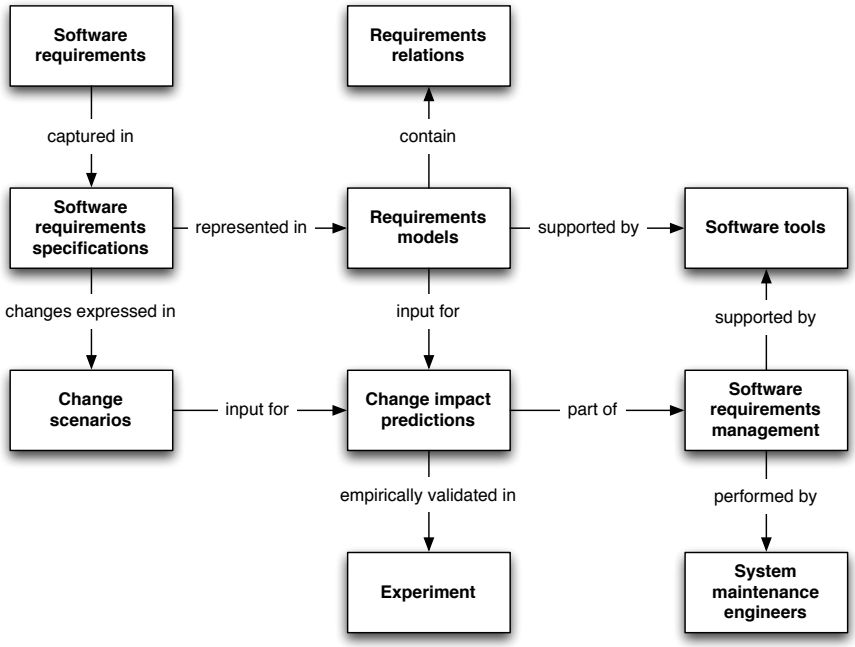


Figure 2: Conceptual framework for background and relevant work

The topics in Figure 2 are discussed in the following order. First, core topics to introduce the domain are discussed. These are software requirements, software requirements specifications, software requirements management and system maintenance engineers.

Discussed next are topics that provide specific instrumentation to this research. These are change scenarios, change impact predictions, requirements models and relationships and

software tools. Finally, the topic of experiments is raised with a discussion of the investigated approach, alternative validation methods and related experiments.

2.2. Software requirements

The term requirement is not used in a consistent way in the software industry [55]. This research uses the definition provided by the IEEE Standard Computer Dictionary [58]:

1. A condition or capability needed by a user to solve a problem or achieve an objective;
2. A condition or capability by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents;
3. A documented representation of a condition or capability as in 1 or 2.

Requirements are part of a software requirements specification [59]. Knowledge about the characteristics of requirements is thus necessary to understand software requirements specifications as a greater whole.

Requirements can differ in structure, contents and style. The following paragraphs describe related work on these characterizations.

Requirements structure

Requirements are often written in natural language but may be written in a particular requirements specification language. When expressed in specification language, they may additionally retain their description in natural language. Representation tools can describe the external behavior of a requirement in terms of some abstract notion [59]. Note that TRIC does not describe external behavior of requirements but the relationships between requirements, and thus is not a representation tool.

Requirements may be uniquely identified if they have a unique name or reference number, which facilitates forward traceability. They may facilitate backwards traceability if they explicitly reference their source in earlier documents [59].

Some requirements descriptions use the phrase “to be determined” or “TBD”. In that case, the description can state the conditions causing this status, what must be done to eliminate it, who is responsible for the elimination and when it should be eliminated [59].

Requirements can be ranked for importance or stability. Stability can be expressed in terms of the number of expected changes to any requirement based on experience of forthcoming

events [59]. Importance can refer to the level of necessity or priority [39]. One widely used technique for ranking importance or necessity is called MoSCoW, which defines "Must Have", "Should Have", "Could Have" and "Won't Have requirement" rankings [7]. Any other scale may be developed [39], one example being the "Essential", "Conditional" and "Optional" scale that is presented in IEEE Std 830-1998 [59]. Priorities are usually used as weighting factor and can likewise be measured on any scale [39].

A highly common way to express requirements is using the feature requirement style [39]. Example requirements expressed using this style are the following:

> **R1:** The product shall be able to record that a room is occupied for repair in a specified period.
>
> **R2:** The product shall be able to show and print a suggestion for staffing during the next two weeks based on historical room occupation. The supplier shall specify the calculation details.
>
> **R3:** The product shall be able to run in a mode where rooms are not booked by room number, but only by room type. Actual room allocation is not done until check-in.
>
> **R4:** The product shall be able to print out a sheet in which room allocation for each room booked under one stay.

Note that the requirements are described in natural language and have a unique identifier, and are not ranked or expressed in a specification language. Other styles for expressing requirements are discussed later.

Requirements contents

Requirements can be classified depending on the kind of condition or capability that they describe. The classification is not standardized, but it is generally agreed that functional requirements specify a function that a system or system component must be able to perform [59] and that non-functional requirements specify how well the system should perform its intended functions [39].

Additional classes of requirements can be found in the literature. For example, Lauesen [39] also discusses the following:

- **Data requirements:** data that the system should input, output and store internally.
- **Other deliverables:** required deliverables besides hardware and software, such as documentation and specified services.

- **Managerial requirements:** when deliverables will be delivered, the price and when to pay it, how to check that everything is working, what happens if things go wrong, etc. IEEE Std 830-1998 [59] also recognizes these, but maintains that these should not be provided as specific requirements but rather as a separate part in software requirements specifications.

Sommerville [55] also discerns domain requirements that come from the application domain of the system and reflect characteristics and constraints of that domain. These requirements may be either functional or non-functional and thus are not truly a separate class of requirements with respect to their contents. For this reason, this research disregards domain requirements as a separate classification.

Requirements styles

Requirements may be expressed in a variety of styles depending on the classification of a requirement. Lauesen [39] describes over 25 styles, including the previously illustrated feature list style. Each style has its own advantages and disadvantages. Indeed, there is no best style to express requirements. TRIC only supports the feature list style.

2.3. Software requirements specifications

Software requirements specifications are documentation of the essential requirements of the software and its external interfaces [12]. Documented representations of specific requirements in various styles are but one part of it, as it typically also contains other elements [55].

The parts of software requirements specifications are not standardized, although several guidelines exist, including IEEE Std 830-1998 [59], the Volere template [51] and those provided by Lauesen [39] and Sommerville [55].

This research uses IEEE Std 830-1998 as leading guideline for two reasons. First, because it contains recognized quality criteria for software requirements specifications that may serve as useful metrics. Second, because it is aligned with ISO/IEC 12207 [29], an industrial standard in information technology for for software life cycle processes, which is useful in the context of change impact analysis and the QuadREAD Project.

IEEE Std 830-1998 discusses essential parts of a software requirements specification and provides several example templates on an informative basis [59]. The essential parts are captured in a prototype software requirements specification outline. See Figure 3.

Table of Contents

1. Introduction
    1. Purpose
    2. Scope
    3. Definitions, acronyms and abbreviations
    4. References
    5. Overview
2. Overall description
    1. Product perspective
    2. Product functions
    3. User characteristics
    4. Constraints
    5. Assumptions and dependencies
3. Specific requirements

Appendixes

Index

Figure 3: Prototype software requirements specification outline [59]

Other guidelines generally agree with the parts that a software requirements specification should contain. Differences lie in the ordering and composition of parts. For example, the Volere template dictates to have separate parts for functional and non-functional requirements [51] while IEEE Std 830-1998 makes no such distinction in its description of specific requirements [59]. In all guidelines, the parts containing requirements representations are separate from parts containing domain and product insights.

2.4. Software requirements management

Requirements evolution both during the requirements engineering process and after a system has gone into service is inevitable. Software requirements management is the process of understanding and controlling changes to requirements for software products [55].

Requirements management should be done by a change control board with the authority to decide on changes to be made or not. The basic change cycle is as follows [39]:

1. **Reporting:** a requirements issue is reported to the change control board
2. **Analysis:** the issue is analyzed together with other issues
3. **Decision:** evaluate the issue and plan what to do with it

4. **Reply:** report the decision to the source and other people impacted by it
5. **Carry out the decision:** execute the plan

This research is interested in the quality of change impact predictions. These predictions are a result of change impact analysis in the analysis phase.

2.5. System maintenance engineers

Change impact analysis is performed by system maintenance engineers, which are a particular type of requirements engineer. System maintenance engineers use the requirements to understand the system and the relationships between its parts [55]. Based on this understanding, they predict the impact that a requested change in a particular requirement will have on other requirements. Increased understanding about a software requirements specifications helps them to perform this activity effectively [25].

2.6. Change scenarios

Requested changes can take the form of change scenarios, which describe possible change situations that will cause the maintenance organization to perform changes in the software and/or hardware [11].

Scenarios should define very concrete situations. They may be assigned an associated weight or probability of occurrence within a certain time. For example, a change scenario could be: "Due to a new type of pump, the pump interface must be changed from duty cycle into a digital interface, with a set value in kP (kilo Pascal)." [11]

Several scenario-based methods have been proposed to evaluate software architectures with respect do desired quality attributes such as maintainability, performance, and so on [8]. As a systematic literature review with the query *("change scenario" OR "change scenarios") AND software* on the Scopus and Web of Science databases turned out, there has been little focus on change scenarios themselves.

Generally, change scenarios may be elicited by interviewing stakeholders. Here, it is important to interview different stakeholders to capture scenarios from different perspectives. This adds to the diversity of change scenarios. It is also observed that engineers have a certain bias in proposing scenarios that have already been considered in the design of the system [38].

One downside to eliciting change scenarios from stakeholders is that most suggested scenarios relate to issues very close in time, e.g. anticipated changes. To address this issue, it may be

helpful to have some organizing principle while eliciting scenarios. This principle may take the form of a, possibly hierarchical, classification of scenarios to draw from [38].

Evaluating scenarios is hard. Ripple effects are hard to identify since they are the result of details not yet known at the level in which the scenarios are expressed [38]. Indeed, architecture details are not known at the requirements level in this research.

In summary, there is little theory on change scenarios. That and problems regarding their representativeness and validity pose weaknesses to methodologies that depend on them [11].

2.7. Change impact predictions

Change impact predictions enumerate the set of objects estimated to be affected by the change impact analysis method. Change impact analysis is the identification of potential consequences of a change, or estimating what needs to be modified to accomplish a change [6].

A number of sets can be recognized in the context of change impact prediction [2]. See Table 5.

Set	Abbreviation	Description
System	-	Set of all objects under consideration.
Estimated Impact Set	EIS	Set of objects that are estimated to be affected by the change.
Actual Impact Set	AIS	Set of objects that were actually modified as the result of performing the change.
False Positive Impact Set	FPIS	Set of objects that were estimated by the change impact analysis to be affected, but were not affected during performing the change.
Discovered Impact Set	DIS	Set of objects that were not estimated by the change impact analysis to be affected, but were affected during performing the change.

Table 5: Change impact prediction sets [2]

Table 5 shows that the Estimated Impact Set, which is the change impact prediction, may not be equal to the Actual Impact Set. See Figure 11.

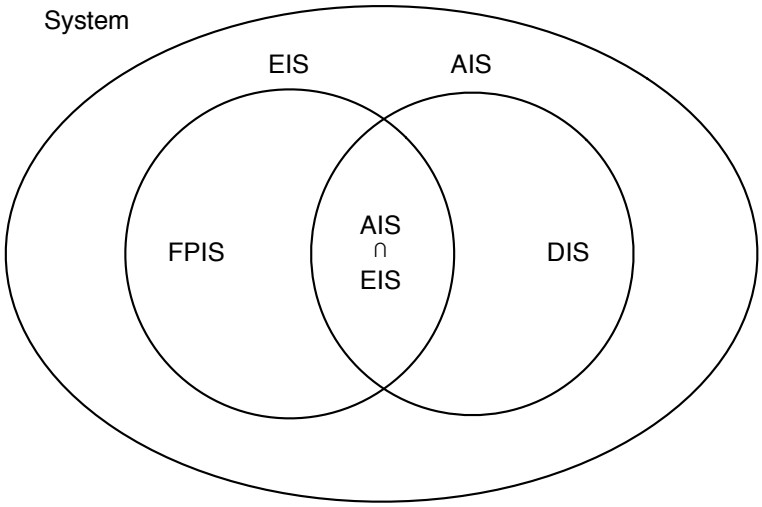


Figure 11: Change impact prediction sets [6]

The Venn diagram in Figure 11 gives a visual representation of the change impact prediction sets in Table 5. In particular, change impact predictions may falsely estimate objects to change (False Positive Impact Set) or falsely estimate objects to not change (Discovered Impact Set). This leads to the thought that there is a quality attribute to change impact predictions.

Quality of change impact predictions

The extent to which the Estimated Impact Set equals the Actual Impact Set is an indication of change impact prediction quality. An object estimated to change may indeed change or it may not; an object actually changed may have been estimated or it may not have been. This may be captured using a binary classifier; see the so-called *confusion matrix* in Table 6 [20].

		Actual Impact	
		Changed	Not changed
Estimated Impact	Changed	True Positive	False Positive
	Not changed	False Negative	True Negative

Table 6: Confusion matrix [20]

Binary classifiers are also used in the domain of information retrieval. Metrics from this domain may be used to measure the quality of change impact predictions [2]. See Table 7.

Metric	Equation	Also known as
Recall	$\frac{\lvert EIS \cap AIS \rvert}{\lvert AIS \rvert}$	Hit rate, sensitivity, true positive rate
Precision	$\frac{\lvert EIS \cap AIS \rvert}{\lvert EIS \rvert}$	Positive predictive value
Fallout	$\frac{\lvert FPIS \rvert}{\lvert System \rvert - \lvert AIS \rvert}$	False alarm rate, false positive rate

Table 7: Change impact prediction quality metrics [2]

A popular measure that combines precision and recall is the weighted harmonic mean of precision and recall, also known as the F_1 measure because recall and precision are evenly weighted [2]. See Equation 1.

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

Equation 1: F_1 measure [2]

Measures such as $F_{0,5}$ and F_2 weigh either the precision or recall double and can be used if either precision or recall is more important than the other in a certain situation [2]. The F_1-measure is used the most and is henceforth referred to as the F-*measure*. Results on the *F*-measure are referred to as F-*scores*.

Another quality attribute of change impact predictions is the effort that it takes. While the *F*-measure can be regarded as a quality measure of change impact prediction products, the measurement of change impact prediction process effort is left to human judgement [12]. Time is one plausible metric [44] to measure effort but does not represent it fully. For example, a group using TRIC may take much longer looking at visualization output, while viewing the visualization may take only one mouse-click.

Visualization techniques

Popular methods to visualize measurements on these metrics are the Receiver Operating Characteristic, or ROC curve, and Precision-Recall graph [2]. The ROC curve is a graphical plot of the recall versus fallout. The Precision-Recall graph is exactly that: a graphical plot of the precision versus recall. See Figure 12 and Figure 13.

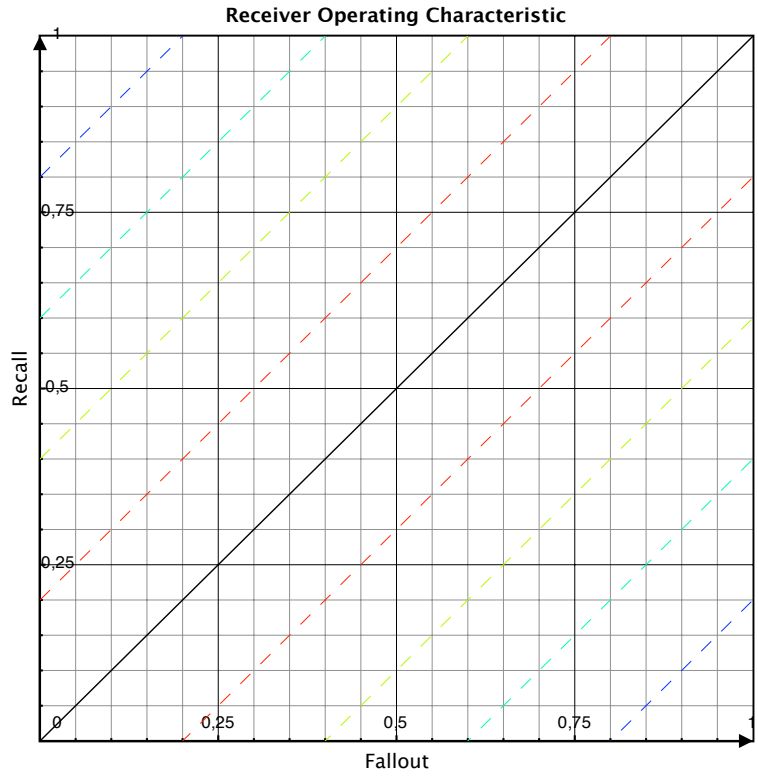


Figure 12: Receiver Operating Characteristic

Figure 13 shows a ROC curve of change impact predictions made by different people for the same change scenario. The X axis displays their fallout scores on a scale of 0 (no false positives) to 1 (all possible false positives). The Y axis displays their recall scores on a scale of 0 (no true positives) to 1 (all possible true positives). The circles represent the scores of the individual predictions.

In a ROC curve the black diagonal line from (0, 0) through (1, 1) is called *the line of no discrimination* [2]. Scores along this line are effectively random guesses: the estimations were comprised of an equal number of true positives and false positives.

The diagonal lines that are parallel to it are isocost lines. Scores along these lines have an equal cost of false negatives versus false positives. It is thus desirable to maximize recall and minimize fallout, placing scores as far away as possible from the line of no discrimination in northwestern direction at a 90° angle [2].

Scores southeast of the line of no discrimination are called *perverse scores* because they are worse than those of random predictions. Such scores may be transformed into better-than-random scores by inverting their Estimated Impact Sets, effectively mirroring the score over the line of no discrimination [2].

The gradient of the isocost lines in this ROC curve are at 45°, indicating that the cost of false negatives versus false positives is equal, which is a common assumption. Like the F_1-score, an emphasis may be placed on either false negatives or false positives if appropriate to the situation, in which case the gradient will change [2].

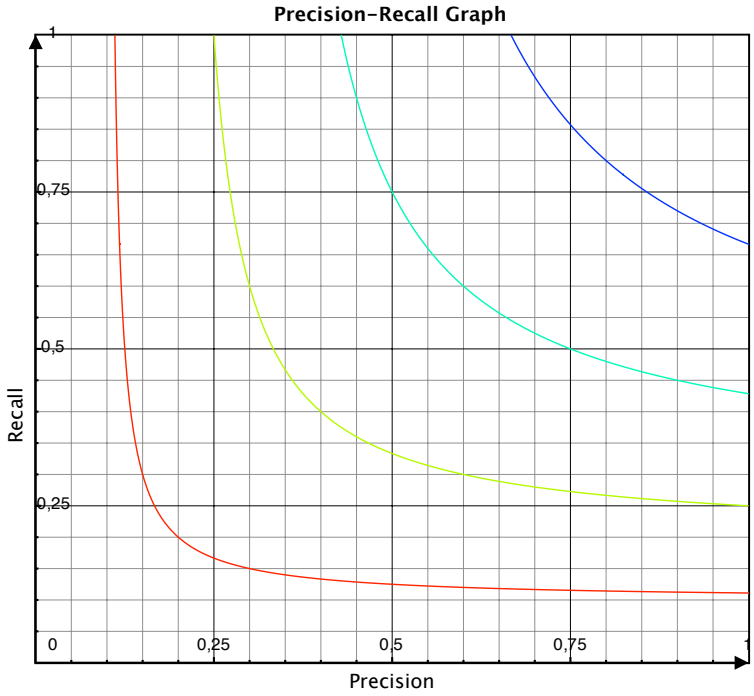


Figure 13: Precision-Recall graph

Figure 13 shows a Precision-Recall graph of change impact predictions made by different people for the same change scenario. The X axis displays their recall scores on a scale of 0 (no true positives) to 1 (all possible true positives). The Y axis displays their precision scores on a scale of 0 (estimation contained no true positives) to 1 (estimation contained only true positives). The circles represent the scores of the individual predictions.

The isometric lines show boundaries of *F*-scores, from southwest to northeast: 0,2; 0,4; 0,6 and 0,8. It is thus desirable to maximize both precision and recall [2].

ROC curves are commonly used to present results for binary decision problems. However, when dealing with highly skewed datasets, Precision-Recall graphs give a more informative picture [15].

2.8. Requirements models and relations

One requirement in a software requirements specification may be related to one or more other requirements in that specification. Relationships can be of a certain type that more precisely defines how the requirements are related. Using imprecise relationship types may produce deficient results in requirements engineering. For example, during change impact analysis requirements engineers may have to manually analyze all requirements in a software requirements specification. This will lead to more costly change implementation [25].

Different vocabularies with types of relationships exist. For example, IBM Rational RequisitePro defines *traceTo* and *traceFrom* [27]. These only indicate the direction in the relationship and are thus very generic [25]. Another example is OMG SysML, which defines *contain, copy, derive* [47] and includes the standard *refine* UML stereotype [67]. These are defined only informally in natural language and therefore imprecise.

The QuadREAD Project has contributed a requirements metamodel with formal relationship types; see Figure 4. The formalization is based on first-order logic and is used for consistency checking of relationships and inferencing. Consistency checking is the activity to identify the relationship whose existence causes a contradiction. Inferencing is the activity of deriving new relationships based solely on the relationships which a requirements engineer has already specified [25].

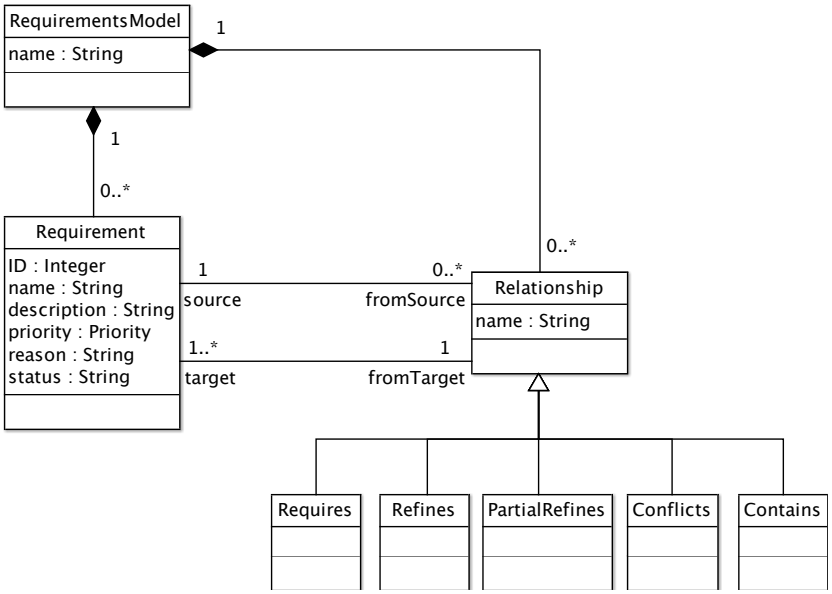


Figure 4: Requirements metamodel [25]

In Figure 4, a software requirements specification is composed of any number of requirements and relationships. Each relationships has one of five types according to the following informal definitions [25] and illustrations [24]:

- **Requires relationship.** A requirement R_1 *requires* a requirement R_2 if R_1 is fulfilled only when R_2 is fulfilled. The requirement can be seen as a precondition for the requiring requirement [65]. See Figure 5.

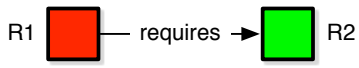


Figure 5: Requires relationship [24]

- **Refines relationship.** A requirement R_1 *refines* a requirement R_2 if R_1 is derived from R_2 by adding more details to its properties. The refined requirement can be seen as an abstraction of the detailed requirements [14] [62]. See Figure 6.

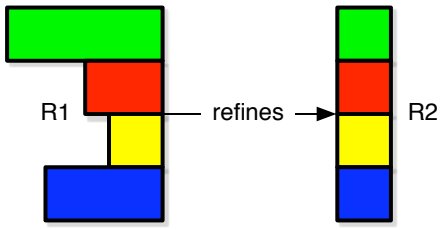


Figure 6: Refines relationship [24]

- **Partially refines relationship.** A requirement R_1 *partially refines* a requirement R_2 if R_1 is derived from R_2 by adding more details to parts of R_2 and excluding the unrefined parts of R_2. This relationship can be described as a special combination of decomposition and refinement [62]. See Figure 7.

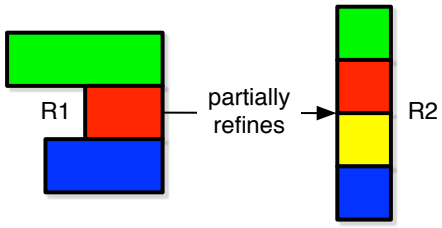


Figure 7: Partially refines relationship [24]

- **Contains relationship.** A requirement R_1 *contains* requirements R_2...R_n if R_2...R_n are parts of the whole R_1 (part-whole hierarchy). This relationship enables a complex requirement to be decomposed into parts [47]. See Figures 8 and 9.

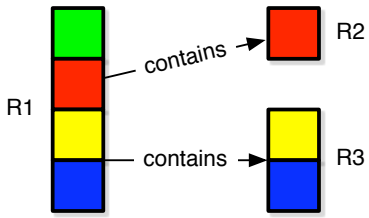


Figure 8: Contains relationship: partial decomposition [24]

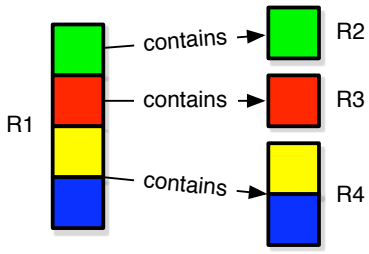


Figure 9: Contains relationship: complete decomposition [24]

- **Conflicts relationship.** A requirement R_1 *conflicts with* a requirement R_2 if the fulfillment of R_1 excludes the fulfillment of R_2 and vice versa [63]. There may be conflicts among multiple requirements that are non-conflicting pairwise. See Figure 10.

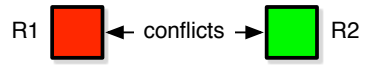


Figure 10: Conflicts relationship [24]

The requirements metamodel also has formal definitions of these relationships and proofs for the consistency checking and inferencing capabilities. They are omitted here because the informal definitions convey enough information to a practitioner to apply them in software requirements management. This was confirmed in the QuadREAD Advisory Board Meeting with the project partners on June 4, 2009.

2.9. Software tools

This research investigates three software tools that support requirements management at different levels of intelligence and maturity:

- Microsoft Excel is a popular general-purpose spreadsheet application.
- IBM Rational RequisitePro is a dedicated requirements management application that is well-known in the industry.

- TRIC is a prototype requirements management tool with support for the formal requirements relationship types.

The following paragraphs discuss features of these software tools, present a classification scheme for comparison and finally compare the software tools.

Features

Requirements management tools may support many features and not all will apply to this research. The TRIC feature list contains the following [25]:

- Management of requirements: the creation, updating, viewing and deletion of requirements. The software tool should support this in a graphical fashion for ease of use.

- Management of requirements relationships: the creation, updating, viewing and deletion of relations between requirements. This effectively adds traceability support to the software tool, which has been shown to be important to practicing effective change management.

- Traceability matrix visualization: the display of related requirements in an n×n matrix. This is a common way to visualize traceability in a software requirements specification and may be used to propagate changes from one requirement to related requirements, which is useful in change impact analysis [39].

- Automated reasoning based on requirements relationships, such as:

    - Displaying inconsistencies: the automated detection and visualization of inconsistencies in the requirements. This will only be possible if the software tool supports management of requirements relationships and their relationship types carry semantics.

    - Displaying inferred relationships: the automatic detection and visualization of requirements relationships that were determined to exist based on given requirements relationships. In its simplest form, this may be done by applying transitivity. More advanced tools can apply more advanced reasoning if their relationship types carry semantics.

    - Explaining reasoning results: the visualization of the process of consistency checking and inferencing. This provides additional knowledge while practicing change management, which can make it more effective.

Microsoft Excel

Microsoft Excel is a popular general-purpose spreadsheet application. Although it is not a dedicated requirements management tool, it can be used to keep a list of requirements and relate them to each other, for example using a traceability matrix. See Figure 14.

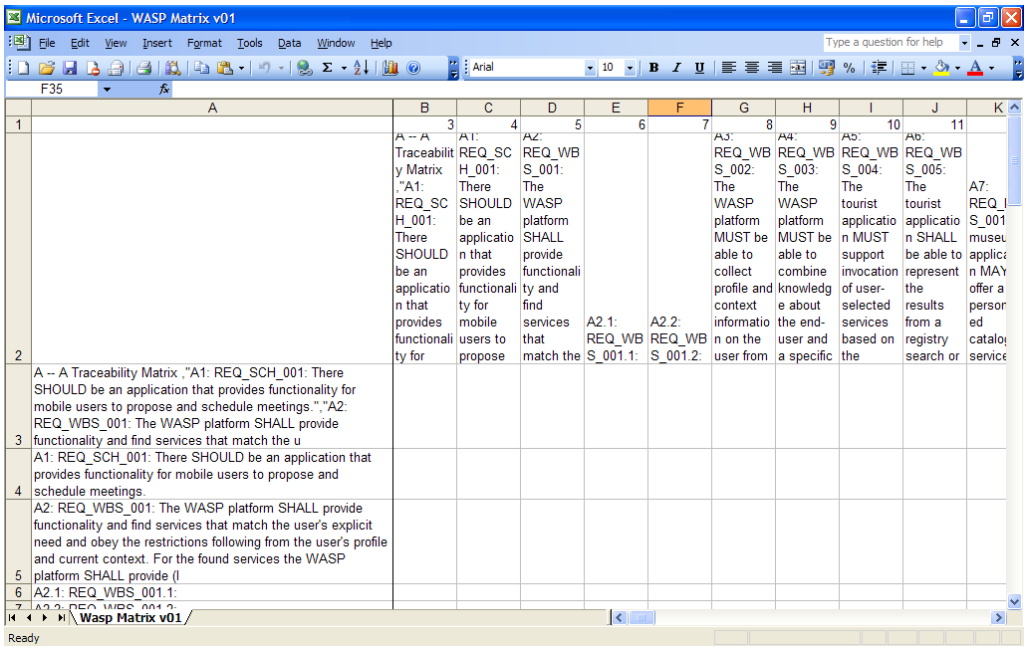


Figure 14: Traceability matrix in Microsoft Excel

Because Microsoft Excel is not a dedicated requirements management tool, performing requirements management with it will largely be an ad-hoc activity. It carries no semantics and cannot perform inferencing or consistency checking.

This research uses Microsoft Excel 2003.

IBM Rational RequisitePro

Rational RequisitePro is a requirements management and use-case writing tool, intended to help improve communication and traceability, enhance collaborative development, and integrate requirements throughout the lifecycle [27].

This research uses IBM Rational RequisitePro version 7.1. As a mature requirements management tool, it offers many features. The following features are of interest to this research regarding traceability and change management:

- **Managing requirements.** Requirements can be created, updated, viewed and deleted using a GUI. See Figure 15.

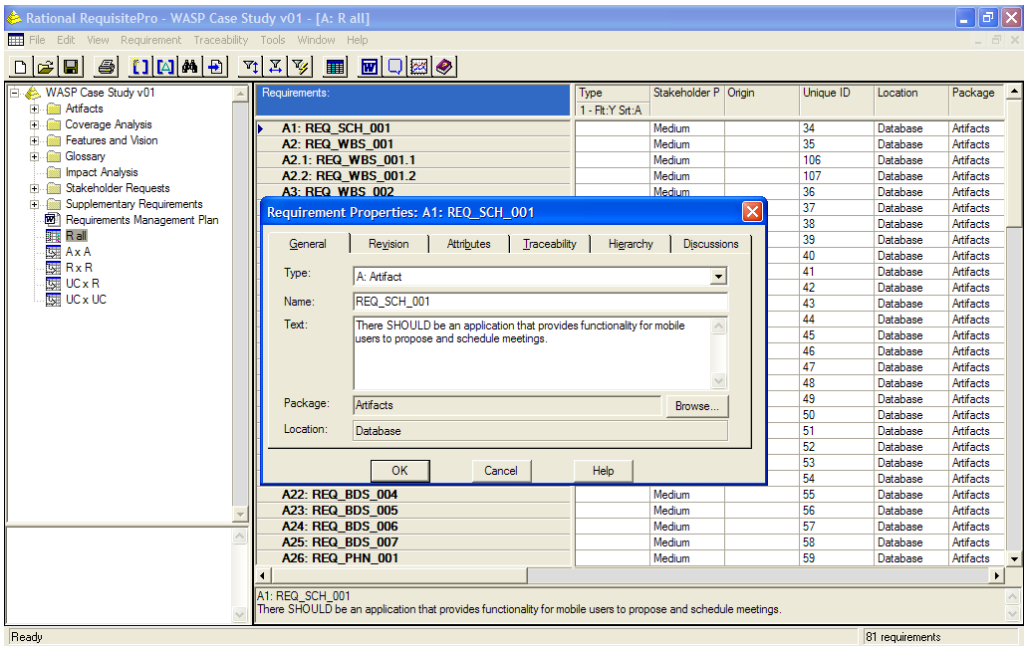


Figure 15: GUI for managing requirements and relations in IBM Rational RequisitePro

- **Managing requirements relationships.** Relationships can be created, updated, viewed and deleted using the requirement management GUI in Figure 15 or the traceability matrix in Figure 16.

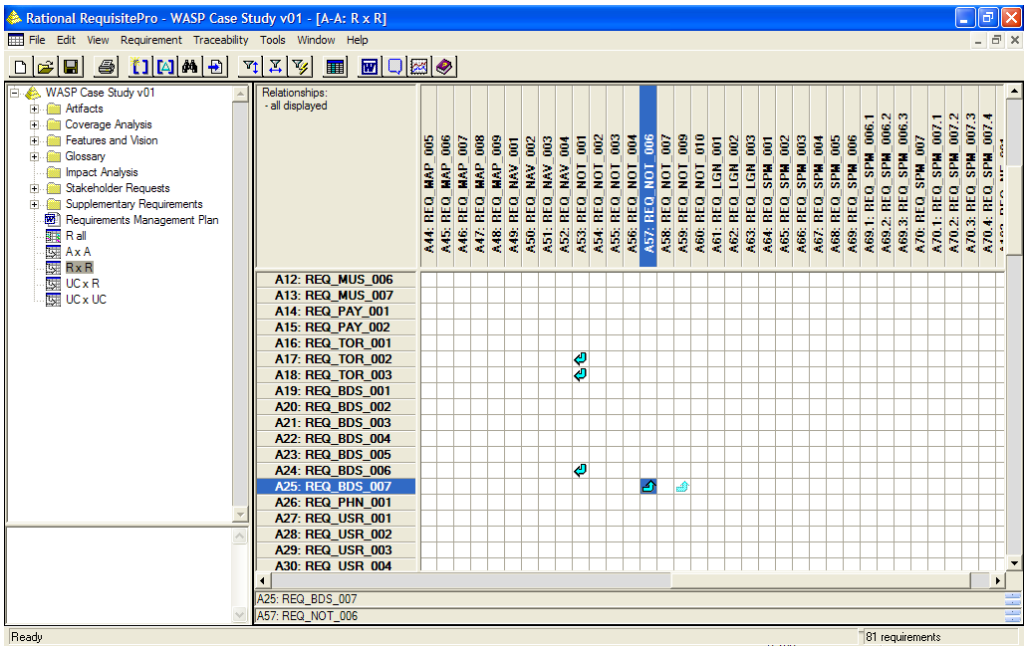


Figure 16: Traceability matrix in IBM Rational RequisitePro

- **Displaying suspect relations.** Relationships deemed suspect based on transitivity are highlighted. In Figure 16, a relationship is suspected between A25 and A59.

TRIC

A prototype software tool was developed for requirements modeling with support for the formal requirements relationships that were defined in the requirements metamodel. This software tool is called TRIC: Tool for Requirements Inferencing and Consistency Checking [64].

This research uses TRIC version 1.1.0. It supports the following features [25]:

- **Managing requirements.** Requirements can be created, updated, viewed and deleted using a GUI. See Figure 17.

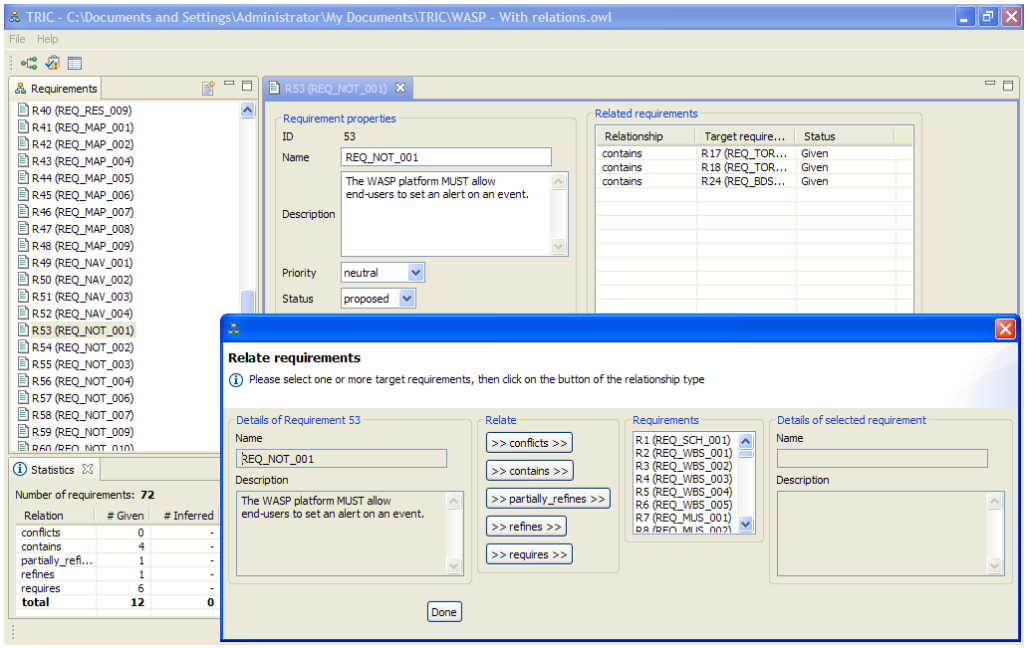


Figure 17: GUI for managing requirements and relations in TRIC

- **Managing requirements relationships.** Relationships can be created, updated, viewed and deleted using the requirement management GUI in Figure 17 or the traceability matrix in Figure 18.

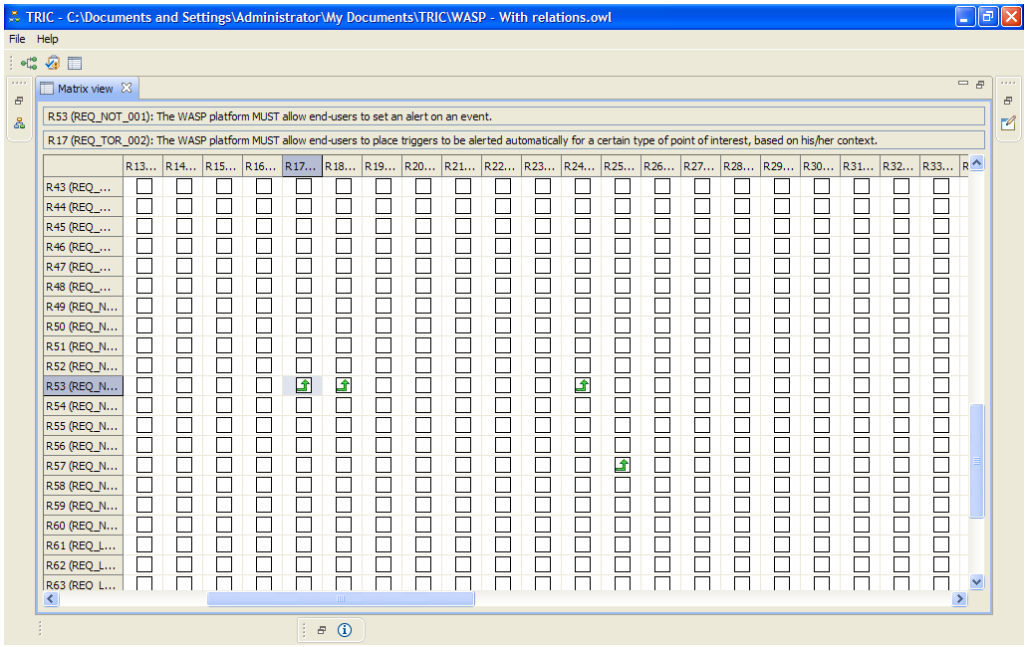


Figure 18: Traceability matrix in TRIC

- **Displaying inconsistencies and inferred relations.** Inferred relationships are highlighted and a list of conflicting requirements is provided. See Figure 19 and Figure 20.

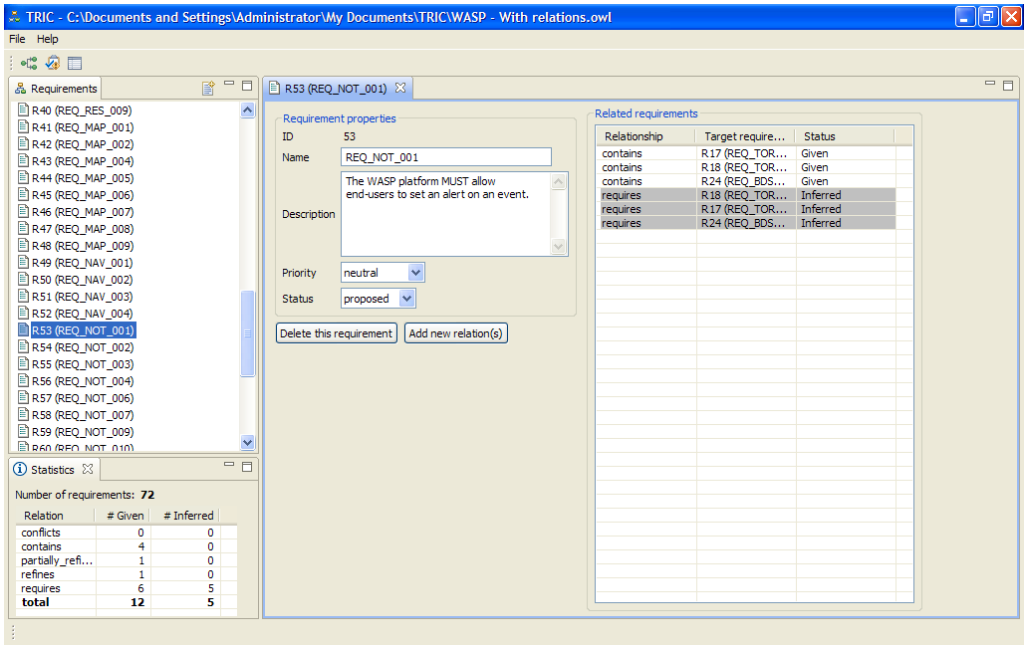


Figure 19: Output of the inferencing activity in TRIC

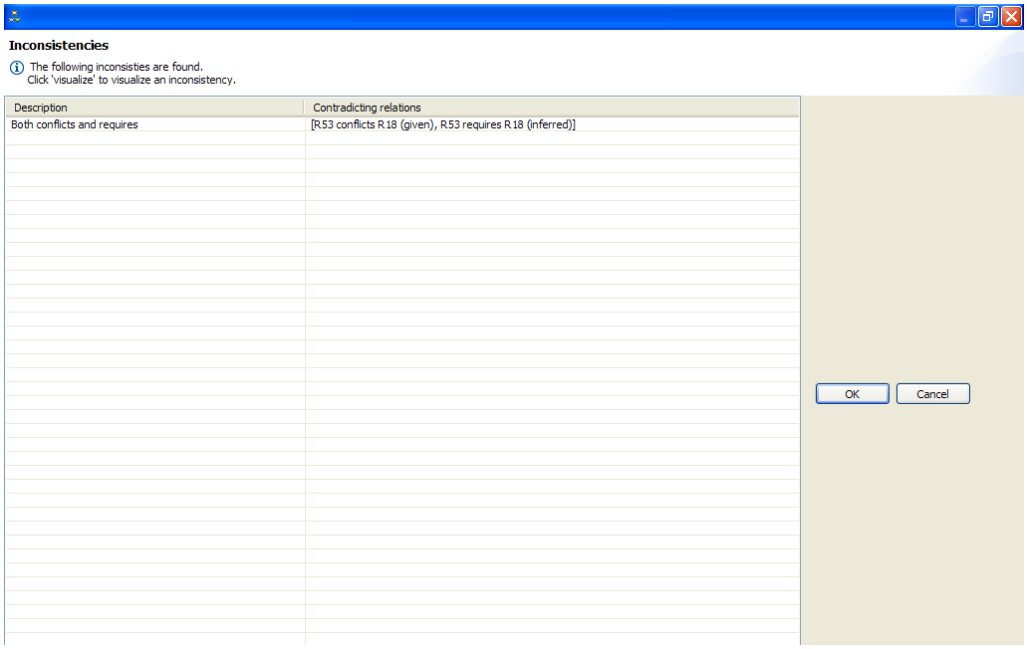


Figure 20: Output of the consistency checking activity in TRIC

- **Displaying results of reasoning.** Sets of requirements and their relations may be expressed as a graph [19]. The given and inferred relationships between requirements are visualized for requirements engineers to interpret. See Figure 21.

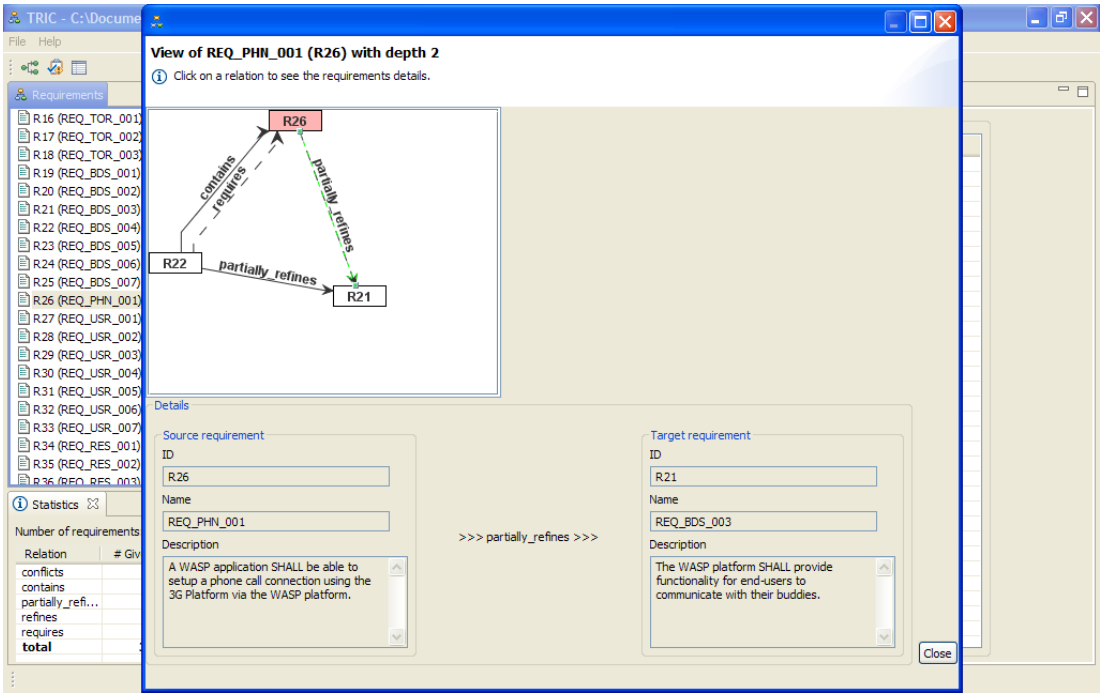


Figure 21: Explanation of the inferred partially refines relationship between R21 and R26 in TRIC

Classification scheme

The described software tools can be compared using the classification scheme. One classification scheme that was developed in the QuadREAD Project compares requirements management tools with support for traceability-based change impact analysis on the following criteria [2]:

- Information model support
- Links detached from artifacts
- Uniquely identifiable reference objects
- Traceability link types
- Assign attributes to links
- Automatic link detection
- Enforce traceability links
- Requirements documentation support
- Tool integration
- Graphical representation traces
- Coverage analysis report
- Impact analysis report

These criteria are then rated on an ordinal scale of "-" (most important aspects of a criterion are missing), "□" (number of important aspects are available, but some are also missing) and "+" (important aspects of a criterion are supported) [2].

There are three reasons why this classification scheme is not suited to this research. First, the software tools that were classified in the original research do not include Microsoft Excel or TRIC. Second, there is no clear operationalization of the criteria, so it is difficult to classify Microsoft Excel or TRIC retroactively. Third, Microsoft Excel is not a requirements management tool with support for traceability-based change impact analysis, so the classification scheme does not apply to it.

An alternative classification scheme can be distilled from the TRIC feature list [25], see Table 8. While this scheme is more limited than the previous scheme and biased towards TRIC, it has the advantage of being easier to identify. Classifications that it produces are not necessarily less precise than those from the previous scheme, because neither scheme has a strong operationalization. It is not the purpose of this research to provide such an operationalization. Rather, an initial comparison is made to show that all three tools are capable of supporting change impact prediction and to offset TRIC's unique features that instigated this research.

This classification scheme is extended with the concepts of software tool intelligence and maturity. Intelligence can refer to the level of reasoning that is supported by a tool. Maturity refers to the development state of a software tool. There is no generally agreed approach to determining the levels of intelligence or maturity, "The complexity of intelligent software and the ambiguities inherent in its interactions with the worlds of human activity frustrate analysis from either the purely mathematical or purely engineering perspectives." [42]

More mature software tools are likely to be more stable, have a more coherent set of features and better interactivity than prototype or immature tools. ISO 13407:1999 provides guidance on human-centered design activities throughout the lifecycle of interactive computer-based systems [28].

The maturity and quality of the interactivity with the intelligence are important to the usability of the software tool. More advanced models can capture knowledge at a greater level of detail [46]. Meanwhile, studies in the domain of web application development have shown that websites with more features but poor presentation are less usable than those with fewer features but with a human-centered design of high quality [57].

Software maintenance engineers can benefit from more advanced capturing of knowledge if the interactivity with that intelligence is of high quality. The codification of software systems into software requirements specifications, understanding of such specifications and act of change impact prediction are all knowledge creation processes. This essentially is externalization of knowledge: the conversion of tacit knowledge into explicit knowledge. Tacit knowledge includes schemata, paradigms, beliefs and viewpoints through that provides "perspectives" that help individuals to perceive and define their world. Explicit knowledge refers to knowledge that is transmittable in formal, systematic language [46].

It is important to build a dialogue between tacit and explicit knowledge. A lack thereof can lead to a superficial interpretation of existing knowledge which has little to do with reality, may fail to embody knowledge in a form that is concrete enough to facilitate further knowledge creation or have little shareability [46]. Experts agree that tacit knowledge is of utmost importance in requirements management, that explicit knowledge is important for communication and shareability, and that the synchronization between tacit and explicit knowledge is challenging. They also add that this is a matter of cost-benefit analysis. If a project is expected to be sensitive to change, highly complex or there is some other stake in the ongoing maintenance, then the benefits of having more detailed knowledge, such as through traceability and semantics, can outweigh the cost of capturing and maintaining the knowledge. See Appendix A.

Comparison

Microsoft Excel, IBM Rational RequisitePro and TRIC may be compared according to the classification scheme described above. See Table 8.

	Microsoft Excel 2003	IBM Rational RequisitePro 7.1	TRIC 1.1.0
Intelligence	Low	Medium	High
Maturity	Mature	Mature	Prototype
Requirements management	Ad-hoc	Supported	Supported
Requirements relations management	Ad-hoc	Yes	Yes
Traceability matrix	Yes	Yes	Yes
Displaying inferred relations	No	Yes, based on transitivity only	Yes, based on reasoning
Displaying inconsistencies	No	No	Yes
Explaining reasoning results	No	No	Yes

Table 8: Comparison of software tools

Table 8 reveals that there is a large commonality between the three tools. While TRIC supports more advanced inferencing, consistency checking and reasoning thanks to its formal requirements relationship types, all three tools at least support management of requirements and requirements relationships and traceability matrix visualization.

Differences in the tools lie in the degree of intelligence, visualization capability and maturity. TRIC supports more advanced reasoning than IBM Rational RequisitePro. And while RequisitePro only supports reasoning based on transitivity, it offers more reasoning than Microsoft Excel, which has no reasoning capabilities at all. It can thus be said that there is an ordinal scale of intelligence, from low to high: Excel, RequisitePro and TRIC.

2.10. Validation approaches

System maintenance engineers will use some software tool to support their requirements management task. Normally this will be a tool such as Microsoft Excel, IBM Rational RequisitePro or alternatives from industry. TRIC is a prototype software tool which is different from these normal tools, because it can reason on requirements relationships. To compare the

TRIC approach to using the classic approach, it requires control over behavioral events for which experimentation is the most appropriate [72].

This experiment will set up a controlled environment in which system maintenance engineers working on a software requirements specification. They will be divided in three groups, each being allowed the use of one of the described software tools, and asked to perform change impact prediction for a set of change scenarios. A complete experimental design is provided in Chapter 3.

An alternative method of validation would be technical action research. Technical action research is a subtype of action research in which a solution technique is investigated by trying it out [68]. Causal interferences about the behavior of human beings are more likely to be valid and enactable when the human beings in question participate in building and testing them [5]. The researcher could help a client while using TRIC, and the client could ask the researcher to help them improve the effectiveness of change management. An analysis of the results can show whether TRIC works in practice.

Action research may be one of the few possible alternatives that is practical given available research resources. The advantage of action research is that it is less vulnerable to Hawthorne effects. A difference is that TRIC will not be validated as-is but as an evolving and co-created prototype, which is not reflective of the current situation [68]. The ideal domain of the action research method is where [10]:

1. The researcher is actively involved, with expected benefit for both researcher and client.
2. The knowledge obtained can be immediately applied. There is not a sense of the detached observer, but that of an active participant wishing to utilize any new knowledge based on an explicit, clear conceptual framework.
3. The research is a cyclical process linking theory and practice.

Given this ideal domain, action research would suit the TRIC solution validation well. TRIC and the requirements metamodel provide an explicit, clear conceptual framework that can immediately be applied. Further, the QuadREAD Project aims to strengthen practical applicability by linking theory and practice [50].

In its broadest sense, action research resembles the act of researchers conducting a highly unstructured field experiment on themselves together with others [9]. They work well when, within a certain population, individual users are the unit of analysis. However, most field experiments will not be able to support the participation of sufficiently large number of popula-

tions to overcome the severity of statistical constraints [72]. This is likely also true for the research in the QuadREAD Project. As elaborated on in Chapter 3, the industry partners do not have high enough availability of people and cases overcome statistical constraints.

Related experiments

A systematic literature review was constructed with the query *("impact prediction" OR "impact analysis") AND ("change" OR "changes") AND ("experiment" OR "experiments" OR "case study" OR "case studies" OR "action research")*. The rationale of this query is that it should retrieve all documents concerning change impact prediction, even if the author shorthands this concept as "impact prediction" or "impact analysis", for which an experimental or real-world research setup was followed.

The Scopus and Web of Science databases were queried. Results that mainly dealt with change analysis at the architecture or implementation level were discarded.

Design recording

An experimental study was conducted in which participants perform impact analysis on alternate forms of design record information. Here, a design record is defined as a collection of information with the purpose to support activities following the development phase [1], which would include traceability artifacts.

The study used a 3×3×3 factorial design featuring a total of 23 subjects, all fourth-year students enrolled in a course on software engineering. The research objects consisted of three versions of an information system for a publishing company, and one change request per information system. The change scenarios were constructed as follows [1]:

- The researchers chose the most realistic change scenarios from a list of suggestions by students in a prior course on software testing.
- A pilot study was conducted to adjust the complexity of maintenance tasks so that they could be completed within a reasonable amount of time, which was specified to be 45 minutes per task. The pilot study revealed that this was too limited and consequently the time limit was extended to 90 minutes per task.
- A change classification of corrective, adaptive and perfective was used. One change scenario per class was selected. References for this classification were not mentioned.

The experimental task was to perform impact analyses for the three change scenarios, each for one information system. The experiment recognized reliability problems when tutoring participants and employed manually-based techniques to avoid effects of unequal training to supporting technologies [1].

The completeness was measured by dividing the number of change impacts correctly predicted by a subject by the actual change impacts [1]. This is equal to the recall metric. The accuracy was measured by dividing the number of change impacts correctly predicted by a subject by the total number of predicted change impacts [1]. This is equal to the precision metric. Finally, the time was measured. It was observed that the participants lacked focus, inducing reliability problems in the data set [1].

Although the results are statistically non-significant, they are said to indicate that design recording approaches slightly differ in work completeness and time to finish but the model dependency descriptor, a certain type of design recording model which describes a software system as a web which integrates different work products of the software life cycle and their mutual relationships, leads to an impact analysis which is the most accurate. Time to finish also increased slightly using the model dependency descriptor. These conclusions were drawn based on post-hoc analyses [1] which were not grounded because underlying assumptions were not met.

These results suggest that design records have the potential to be effective for software maintenance but training and process discipline is needed to make design recording worthwhile [1].

Storymanager

A requirements management tool, the Storymanager, was developed to manage rapidly changing requirements for an eXtreme Programming team. As part of action research, the tool was used in a case project where a mobile application for real markets was produced. The tool was dropped by the team only after two releases. The principle results show that the tool was found to be too difficult to use and that it failed to provide as powerful a visual view as paper-pen board method [31]. This phenomenon was also observed during a think aloud exercise with an assistant professor in Information Systems, see Appendix A.

Trace approach

An approach was introduced that focuses on impact analysis of system requirements changes and that is suited for embedded control systems. The approach is based on a fine-grained trace model. With limited external validity, an empirical study has shown that the approach allows a

more effective impact analysis of changed on embedded systems; the additional information helped in getting a more complete and correct set of predicted change impacts [66].

The research used a two-group design featuring a total of 24 subjects, all master students enrolled in a practical course on software engineering. The research objects consisted of two software documentation subsets and two kinds of development guidelines, totaling 355 pages. They described a building automation system. The experimental task was to perform an impact analysis for the changes in the change requests provided [66]. The design of change scenarios was not specified.

The completeness was measured by dividing the number of change impacts correctly predicted by a subject by the actual change impacts [66]. This is equal to the recall metric. The correctness was measured by dividing the number of change impacts correctly predicted by a subject by the total number of predicted change impacts [66]. This is equal to the precision metric.

traceMAINTAINER

A prototype software tool called traceMAINTAINER was developed to automate traceability maintenance tasks in evolving UML models. A quasi-experimental research design was used to empirically validate the traceMAINTAINER tool. 16 master students following a course on software quality were partitioned in two groups: one with the software tool and the other without the software tool. Each group was tasked to implement three model changes [43].

The research used a two-group design featuring a total of 16 subjects, all Computer Science master students. The research object consisted of UML models on three levels of abstraction for a mail-order system: requirements, design and implementation. The set of traceability relations consists of 214 relations. The experimental task was to perform impact analyses for three change scenarios in 2-3 hours time [43]. The change scenarios were listed but no systematic design scheme was described.

The group with the software tool was provided the software several days in advance. The participants were asked to indicate the number of hours they had spent with it in advance of the experiment [43]. It is unclear if and how this was treated as a covariable, and what the causality between “hours spent” and “tool aptitude” is.

The results of the quasi-experiment were measured in terms of quality, using precision and recall, and in terms of the number of manually performed changes. The research yielded two conclusions with limited generalizability. First, the group using traceMAINTAINER required

significantly fewer manual changes to perform their change management. Second, there was no significant difference between the quality of the change predictions of the two groups [43].

TRIC

TRIC was illustrated using a single fictional case study featuring a course management system [37]. Based on the case study results, it was concluded that TRIC supports a better understanding of mutual dependencies between requirements, but that this result could not be generalized pending a number of industrial and academic case studies with empirical results [25].

2.11. Conclusion

This chapter discussed the relevant work based on a conceptual framework linking the topics together. Requirements and software requirements specifications have been researched previously in much detail, however, the nature of change scenarios has not. This is a concern for the reliability of any empirical research using change scenarios as instrumentation.

It was found that the quality of change impact predictions can be operationalized using the *F*-measure, a metric form the domain of Information Retrieval, and the time taken to complete a prediction. Earlier related experiments and case studies have shown the feasibility of testing techniques for change impact prediction with diverse results. Some concluded a positive impact of more precise traceability on the quality of change impact prediction, while others found no significant differences or even that a negative contribution due to increased tool complexity.

3. Experimental design

3.1. Introduction

This chapter presents the planning for the experiment. It serves as a blueprint for the execution of the experiment and interpretation of its results [73].

The design is based on the research goal and hypotheses that support it. A matching research design is then selected. Following that, the details of the experimental design are discussed, including its parameters, variables, planning, expected participants, objects, instrumentation and procedures for data collection and analysis. Finally, the validity of the experimental design is evaluated.

3.2. Goal

The goal of this experiment is to analyze the real-world impact of using a software tool with formal requirements relationship types for the purpose of the evaluation of the effectiveness of tools with respect to the quality of change impact predictions.

3.3. Hypothesis

It is hypothesized that using TRIC, a software tool with formal requirements relationship types, will positively impact the quality of change impact predictions. Considering product and process quality separately, the following formal hypotheses are developed:

Hypothesis 1. The *F*-scores of change impact predictions of system maintenance engineers using TRIC will be equal to or less than those from system maintenance engineers not using TRIC. See Hypothesis 1.

$$H_{0,1} : \mu_1 \leq \mu_2$$
$$H_{1,1} : \mu_1 > \mu_2$$

Hypothesis 1: F*-score of change impact predictions*

In Hypothesis 1, μ is the mean *F*-score of change impact predictions. Population 0 consists of system maintenance engineers using TRIC. Population 1 consists of system maintenance engineers not using TRIC.

Hypothesis 2. The time taken to complete change impact predictions of system maintenance engineers using TRIC will be equal to or longer than those from system maintenance engineers not using TRIC. See Hypothesis 2.

$$H_{0,2} : \mu_1 \geq \mu_2$$
$$H_{1,2} : \mu_1 < \mu_2$$

Hypothesis 2: Time taken to complete change impact predictions

In Hypothesis 2, μ is the mean time of change impact predictions as measured in seconds. Population 0 consists of system maintenance engineers using TRIC. Population 1 consists of system maintenance engineers not using TRIC.

The statistical significance level for testing the null hypotheses is 5% (α=0,05). A lower level would be feasible given a large enough sample size, which will not be the case here due to limited time and availability of participants. From previous experiences it is known that most students will not volunteer for a full day. Likewise, experts from industry are too busy to participate a full day even if they are linked to the QuadREAD Project as partner. Ample monetary compensation is not within the budget of this experiment and is conducive to the threat of compensatory inequality [52]. This is further discussed in paragraph 3.7.

3.4. Design

In this research, different groups will be assigned to perform change impact analysis using a different software tool. This research setup involves control over behavioral events during change impact analysis, for which experimental research is the most appropriate [72].

Experimental research has several subtypes, one of them being quasi-experimental research. By definition, quasi-experiments lack random assignment. Assignment to conditions is by means of self-selection or administrator selection [52] such as is the case in our setup. Consequently, quasi-experimentation is the most appropriate research method.

Multiple controlled experimental designs exist, see Table 9.

Validation method	Description	Weaknesses	Strengths
Replicated	Develop multiple versions of the product	• Very expensive • Hawthorne effect	• Can control factors for all treatments
Synthetic	Replicate one factor in laboratory setting	• Scaling up • Interactions among multiple factors	• Can control individual factors • Moderate cost
Dynamic analysis	Execute developed product for performance	• Not related to development method	• Can be automated • Applies to tools
Simulation	Execute product with artificial data	• Data may not represent reality • Not related to development method	• Can be automated • Applies to tools • Evaluation in safe environment

Table 9: Summary of controlled software engineering validation models [73]

This research cannot use a dynamic analysis or simulation design, because these evaluate the product (the object after performing the change) instead of the process (the change analysis itself). This research then follows a synthetic design, which allows controlling the level of tool support while still being feasible for execution within a limited amount of time. See Figure 22.

$$\begin{array}{lll} NR & X_A & O \\ NR & X_B & O \\ NR & X_C & O \end{array}$$

Figure 22: Research design

In Figure 22, *NR* indicates that the research is non-randomized or quasi-experimental. X_A, X_B and X_C correspond to the three software tools. O is the observation of change impact prediction quality, that is, the F-score and time in seconds. This is also known as the basic non-randomized design comparing three treatments [52].

3.5. Parameters

A single real-world software requirements specification will be selected as research object. Predetermined groups of participants will perform change impact prediction on the require-

ments that are present in this specification. The experiment will be conducted at the Faculty of Electrical Engineering, Mathematics and Computer Science.

Ideally, the experiment should be repeated with different real-world software requirements specifications and track specification complexity as a covariable. It is plausible that the complexity of software requirements specifications will be of influence on the results. Each repetition of the experiment should feature another group of participants to rule out any learning effects.

This experiment features only a single software requirements specification and no repetition due to limited time and availability of participants, which is in line with the research contribution to provide a blueprint and an initial execution. Because data will be available for only a single software specification, it will be impossible to measure the influence of software requirements specification complexity on the results. Complexity will be reported for comparison with future studies.

3.6. Variables

Dependent variables

The dependent variables that are measured in the experiment are those that are required to compute the *F*-score, which is a measure of change impact prediction quality. These variables are:

- Size of the Estimated Impact Set
- Size of the False Positive Impact Set
- Size of the Discovered Impact Set

The precision, recall and finally *F*-scores can be computed according to their definitions that were provided in Chapter 2. The required Actual Impact Set is discussed in the paragraph on instrumentation, below.

Independent variables

One independent variable in the experiment is the supplied software tool during change impact analysis. This is measured on a nominal scale: Microsoft Excel, IBM Rational RequisitePro or TRIC.

This nominal scale is preferred over the ordinal scale of software tool intelligence because this research is interested in the impact of TRIC on the quality of change impact predictions as a new technique versus classic techniques, as opposed to the impact of various software tools with the same level of intelligence.

Still, it would be a threat to internal validity to only study the impact of using TRIC versus Microsoft Excel, because such an experimental design would be biased in favor of TRIC. When assuming that requirements relationships play an important role in the results of change impact prediction, it would be logical that a software tool with dedicated support (e.g. TRIC) would score higher than a software tool without such support (e.g. Microsoft Excel). By also studying an industrially accepted tool such as IBM Rational RequisitePro, concerns to validity regarding the bias in tool support are addressed.

Covariate variables

It is expected that a number of participant attributes will be covariate variables influencing the *F*-scores of change impact predictions and the time taken to complete change impact predictions. These are the following:

- **Level of formal education.** Expected participants will be from both academic universities and universities of applied sciences. By their nature, universities of applied science educate their students in a more practical rather than theoretical fashion. It is to be expected that students from academic universities will be more apt at abstract thinking, such as is the case with change analysis in software requirements specifications. This is measured on a nominal scale of "bachelor or lower" or "Bachelor of Science or higher".

- **Nationality.** Expected participants will be mostly Dutch nationals alongside a small number of foreigners. Earlier experiments in software engineering have shown nationality to be a covariate influencing task precision and time [33, 54]. Related to the level of formal education, it is not truly the current nationality that is of interest but the country in which the participant was educated. This is measured on a nominal scale of "in the Netherlands" or "outside of the Netherlands".

- **Gender.** Earlier experiments in software engineering have shown gender to be a covariate influencing task aptitude [33]. This is measured on a nominal scale of "male" or "female".

- **Current educational program.** Expected participants will be currently enrolled in either Computer Science or Business Information Technology. These programs educate the participants differently: Business Information Technology students often work with software

requirements documents as part of project work, while Computer Science students often work with systems on an architecture and implementation level. This is measured on a nominal scale of "Computer Science" or "Business & IT".

- **Completion of a basic requirements engineering course.** Expected participants may have followed a basic course in requirements engineering, introducing them to concepts such as traceability. Having completed such a course is likely to positively influence change impact prediction performance. This is measured on a nominal scale of "Yes" or "No".

- **Completion of an advanced requirements engineering course.** As above.

- **Previous requirements management experience.** Expected participants may have a number of months of experience with requirements management in general. This is measured on a nominal scale of "three months or more" or "less than three months". This split is based on the principle that three months is longer than one quartile academic year, thus ruling out any overlap with the basic requirements engineering courses.

3.7. Planning

The experiment is set to take one afternoon, from 13:45 to 17:30 on June 11, 2009. This strikes a balance between participant availability and focus on one hand, and possibilities for experimentation on the other.

The experiment is planned as follows. See Figure 23.

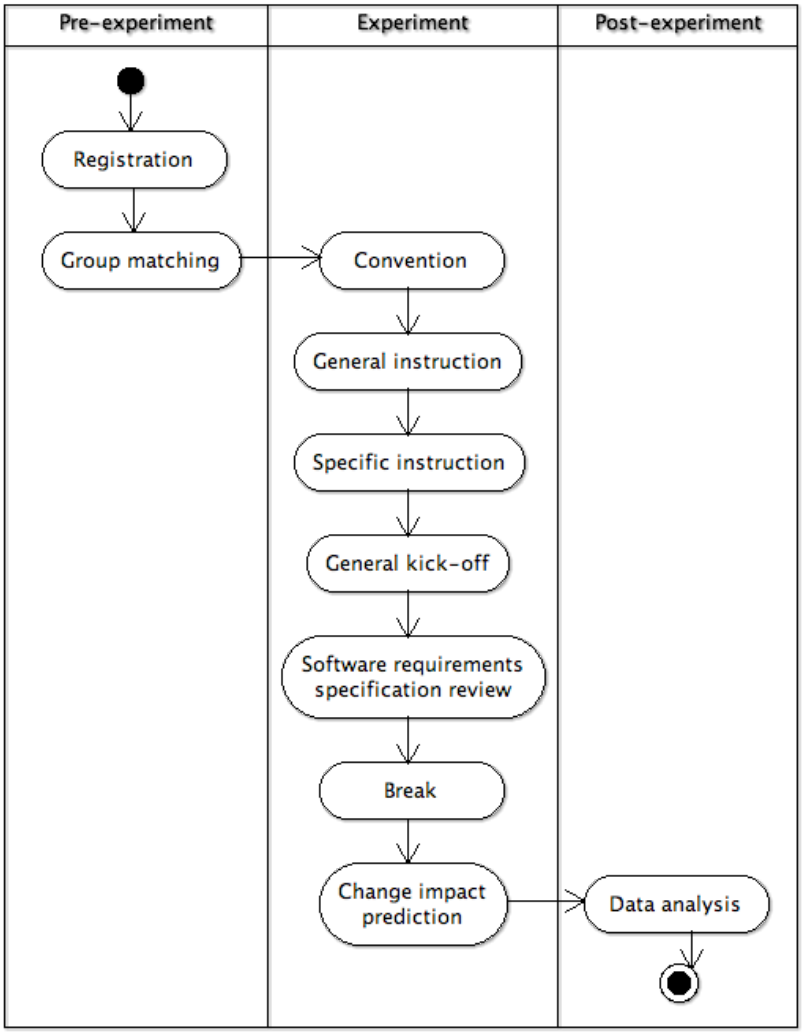


Figure 23: Experiment activity diagram

The experiment is designed to maximize comparability between the groups by only keeping the treatments as equal as possible. Other than the specific instruction and assigned tool for the change impact prediction, all activities and contents are equal.

The following activities can be seen in Figure 23:

1. **Registration (pre-experiment).** Before the start of the experiment, the participants are e-mailed a URL to an online form to register. The e-mail address is gathered from the university course management system. This form collects their name, student number, e-mail address, the covariates described above and if they will bring a laptop. It is noted that the information is used only for the administration of the experiment. The registration closes at 23:59 on the day before the start of the experiment. See Figure 25, Figure 26 and Figure 27.

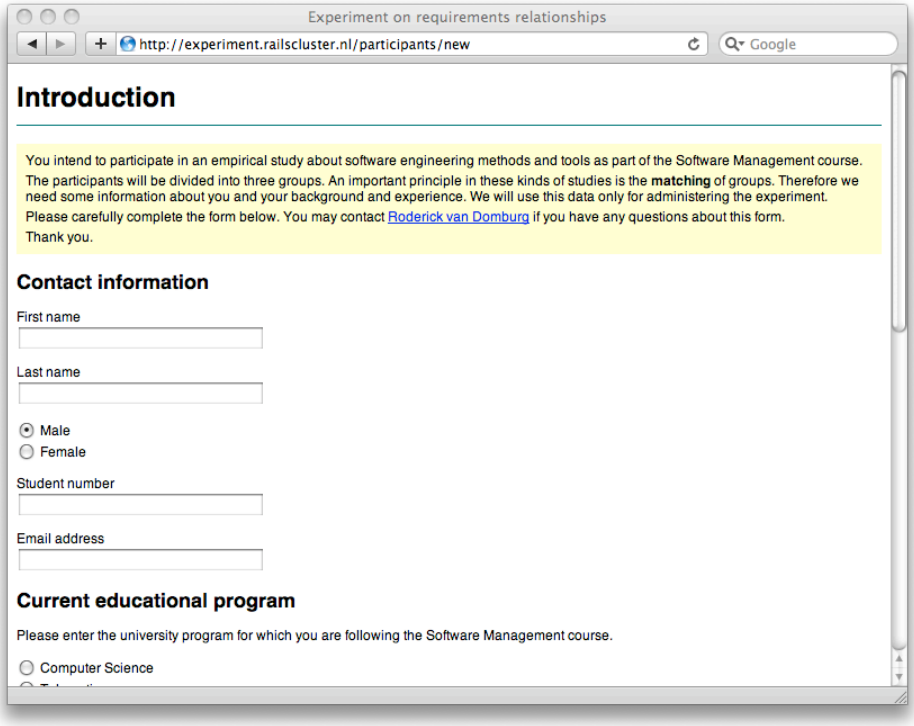


Figure 25: Web application screenshot - Registration (1/3)

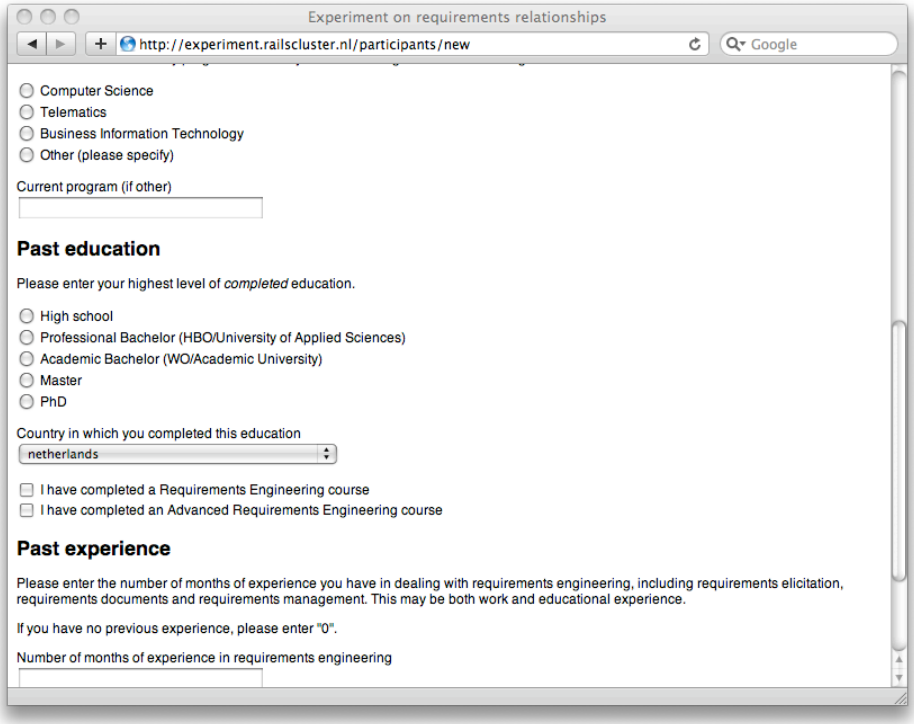


Figure 26: Web application screenshot - Registration (2/3)

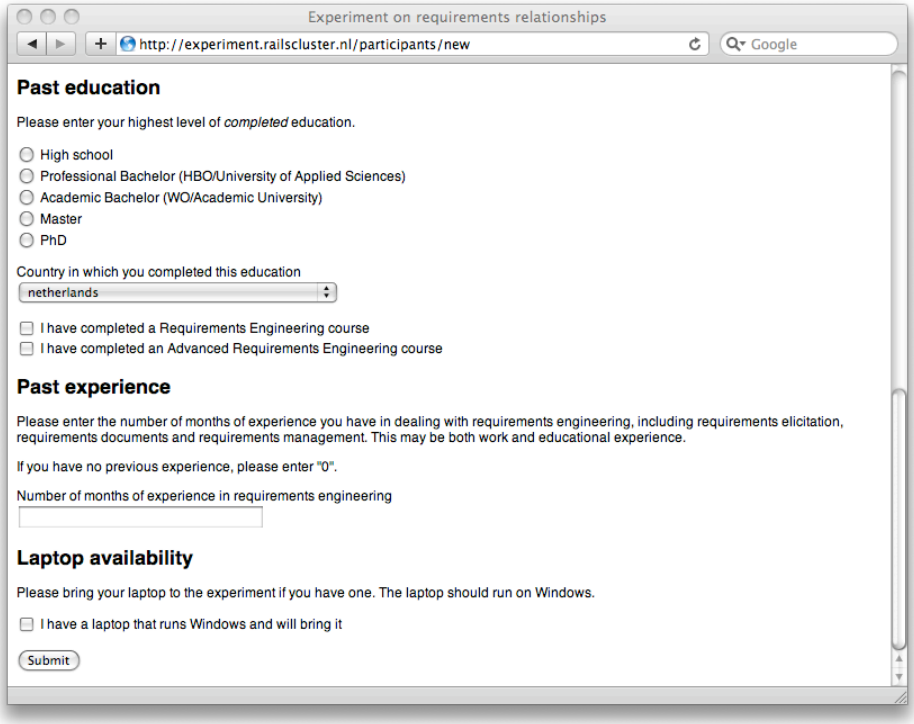


Figure 27: Web application screenshot - Registration (3/3)

2. **Group matching (pre-experiment).** The registered participants are divided over groups. The aim is to have fair group matching: each group should ideally have an equal distribution over the covariate scores. To establish such a group matching, the participants are first randomly divided by ordering their names from A to Z and splitting them into three groups. Expecting an unfair distribution of covariate scores, the groups will then be tuned by manually moving participants from group to group. This is assisted by coding all covariates as "0" or "1".

3. **Convention (15 minutes).** All participants convene in a single room. For each group, the list of participants is read out. Group participants are requested to join their group supervisor who will lead them to their experiment location. There is one separate location per group.

4. **General instruction (15 minutes).** With all groups present on their own location, the supervisor lectures a general instruction. This is led by presentation slides that are equal for all groups. The instruction is practical and geared toward the change management tasks at hand. It introduces the context of change impact analysis, modeling requirements in a traceability matrix and following traces as part of impact estimation to discover re-

lated impacted requirements. It is told that this is going to constitute the tasks of the experiment.

5. **Specific instruction (30 minutes).** Each group supervisor lectures an instruction specific to the software tool that is being used. This instruction is also geared towards performing the change management tasks. To maintain comparability, an equal amount of time is allotted for all these instructions.

    1. The Microsoft Excel group is explained that the spreadsheet application can be used to visualize requirements in a traceability matrix. An example relation is created in the matrix.

    2. The IBM Rational RequisitePro group is explained that it is an application for requirements management with support for change impact analysis using suspected links. It is shown how to perform basic operations such as opening a requirements document and adding, changing or deleting a requirement. It is then shown how to add and inspect relationships, and how to show indirect relationships.

    3. The TRIC group is explained that it is an application for management of requirements relations. The relationship types are shortly introduced using the colored bars from Chapter 2 alongside a one-line example of each relationship type. The results of inferencing are shown. The basic operation of TRIC is demonstrated, including opening a requirements document, adding and deleting relations, using the matrix view, inferencing engine and consistency checker. It is shown how relationships and inconsistencies may be visualized in a graph view.

6. **General kick-off (5 minutes).** Each group supervisor lectures a kick-off presentation containing the prizes, the goal to find the valid impacted requirements in a short time and the URL to an online application that will administer the experiment. The general kick-off is equal for all groups.

7. **Software requirements specification review (60 minutes).** All participants are granted one hour time to individually review the software requirements specification and take any action they deem fit given the upcoming tasks, such as adding notes and relationships in their software tool.

8. **Break (15 minutes).** All participants are offered a break and a soft drink. Each group has its own break location.

9. **Change impact prediction (60 minutes).** All participants are granted one hour to individually perform change impact prediction. This is administered by an online application on which they have an individual user account. See the paragraph on instrumentation, below.

10. **Data analysis (post-experiment).** With all tasks done, the participant is thanked and asked to leave the room in a quiet fashion. The analysis of data and handout of prizes is done after the experiment.

Ideally, the supervisors, who are also lecturers, would have an equal experience in lecturing and equal level of education. It was decided to look for PhD level employees or candidates, because they would have lecturing experience. It turned out to be impossible to find enough of such supervisors with a background in software engineering.

Thus, the group of lecturers was decided to be as follows:

- dr. ir. Klaas van den Berg for the Excel group. He is assistant professor in software engineering with a large lecturing experience. He is involved in the QuadREAD Project and supervisor of Arda Goknil and Roderick van Domburg (see below).

- Arda Goknil, MSc for the RequisitePro group. He is a PhD candidate with the software engineering group researching formal requirements relationships. He should not give the TRIC lecture, because his bias might show. He does not have much lecturing experience.

- Roderick van Domburg, BSc for the TRIC group. He is the master student conducting this research in the QuadREAD Project. He does not have much lecturing experience.

3.8. Participants

Participants will be master students following the Software Management master course at the University of Twente. The experiment is not strictly part of the course and students are encouraged to participate on a voluntary basis. They are offered a present for their participation and promised monetary prizes for the best participants, as measured by the mean F-score over their change impact predictions. Should there be equal mean F-scores, the mean time will be used to make a final decision.

The prizes are as follows. For each software tool group, there is a first prize of € 50 and a second prize of € 30. Everyone is presented with a USB memory stick. Because all participants

principally stand an equal chance of winning the prizes, possible threats to validity due to compensatory inequality are addressed.

Ideally, there would be a group of experts for each group of students. There was no response from the industry partners in the QuadREAD Project inviting them to participate in the experiment. The invitations were sent out by e-mail and previously announced on two QuadREAD Advisory Board meetings.

3.9. Objects

Software requirements specification

The research object is a software requirements specification titled "Requirements for the WASP Application Platform" version 1.0 by the Telematica Instituut [18]. The WASP specification has been used before by the Software Engineering group of the University of Twente. It is a public, real-world requirements specification in the context of context-aware mobile telecommunication services, complete with three scenarios, 16 use cases and 71 requirements. The page count including prefaces is 62. The chapter "Requirements" from the WASP specification has been copied and pasted into Appendix G.

The WASP requirements specification features inter-level tracing from scenarios to use cases and from use cases to scenarios. The requirements are functionally decomposed and ordered in hierarchies. For each function, there is a tree with a calculated tree impurity of 0. Experts rated the WASP specification to be clear and according to best practices, albeit with a lack of a goal. See Appendix A.

The QuadREAD Project members were asked to contribute software requirements specifications from real-world projects in two QuadREAD Advisory Board meetings and one time over direct e-mail. One member responded that he would investigate the possibilities, but was unable to deliver one in the end. The most important reason for this lack of response is cited to be non-disclosure of customer property.

A similar inquiry with two assistant professors in the Information Systems group at the University of Twente also did not deliver any results. The assistant professors explained that the Information Systems group usually operates between the business and requirements layers, where they are more concerned with requirements elicitation than with requirements management. Consequently they did not have any software requirements specifications on file other than those in Lauesen [39].

The WASP requirements specification was chosen over the examples in Lauesen [39] because the latter ones were only excerpts. Also, they were unambiguous due to the availability of a glossary with a well-defined grammar. Such glossaries would have made change impact prediction much easier, which could make the hypothesized advantages of using TRIC invisible in the experiment.

Change scenarios

It is difficult to establish proper change scenarios due to a lack of theory on what a proper change scenario should be. Further, it is unsure if such theory would be beneficial to the experiment. First, from a standpoint of representativeness, real-world change scenarios also lack formality. Second, from a standpoint of ambiguity, an unambiguous change scenario will likely cause our experiment to find no differences between participants' change impact predictions because all participants understood the change scenario equally well.

It is decided to create scenarios so that they cover a range of change scenario cases. Five separate cases can be discerned in the theory on formal requirements relationships [25]. See Table 10.

Case	Tasks
Add part	1
Remove part	2, 4
Add detail to part	3
Add whole	-
Remove whole	5

Table 10: Change scenario cases and tasks

Table 10 shows the five change scenario cases and matching tasks. The change scenarios are available in Appendix B. For each case, a requirement was selected at random and an appropriate change scenario was created.

No change scenario was created for the "add whole" case because that does not impact other requirements; it may add relationships but not change the related requirement itself. A replacement scenario was created for "remove part". This was convenient because many requirements in the WASP specification have multiple parts.

Due to time constraints, the only discussion on the change scenarios before the experiment was within the supervisory group. This led to the addition of the rationale to the change scenarios. An expert was consulted after the experiment execution. The results of this interview are discussed in Chapter 4.

A challenge in providing multiple change scenarios is the possibility that the effect of performing change impact estimation on one scenario might influence the next. A common design to compensate for this is to provide the change scenarios to participants at random [32], which will also be implemented by this experiment.

3.10. Instrumentation

The experiment locations are equipped with a beamer for the lectures and computers for the participants to work on. The computers will be set up with all three software tools, so that there is room for improvisation if there are technical issues with some computers or an experiment location. The supervisors ensure that participants will only use the software tool that they are entitled to.

All participants are handed out a printout of all slides that were shown to them, a copy of the software requirements specification and a USB memory stick. The memory stick contains the requirements specification in PDF format and a digital requirements document that can be opened with their software tool. It is pre-filled with all requirements but contains no relations. Out of time constraints, the participants are told to treat the introduction, scenario and requirements chapters as leading and the use case chapter as informative.

A web application is created to support the registration of participants, distribution of experiment tasks and collection of data. The web application procedure is as in Figure 28. A use case scenario is described afterwards.

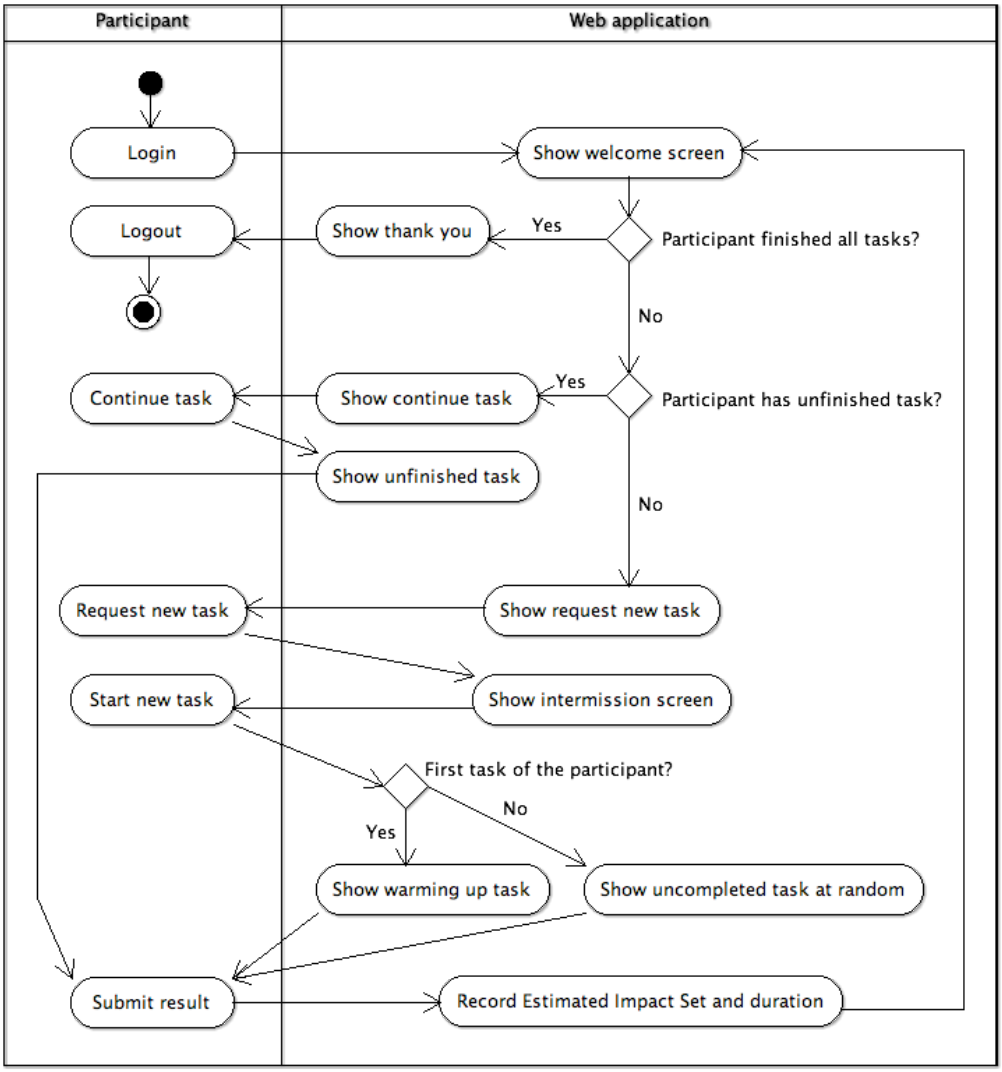


Figure 28: Web application activity diagram

1. Participants log in. A welcome screen is shown, indicating the number of tasks completed and the total number of tasks yet to be completed. It contains a link to request a new task or, if participants had an unfinished task because they logged out or closed their browser prematurely, to continue the unfinished task. See Figure 29 and Figure 30.

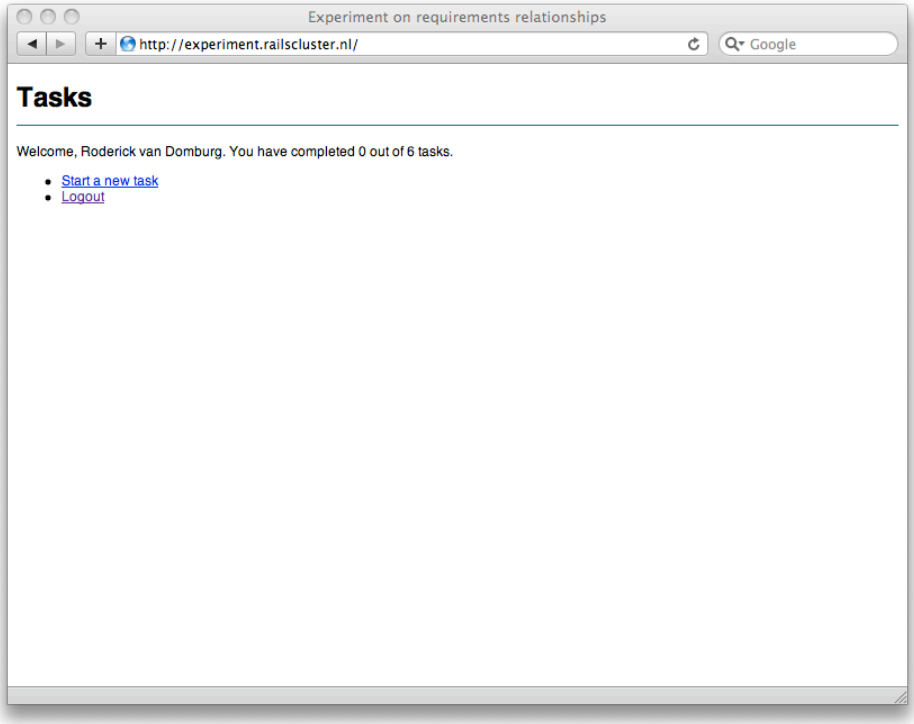


Figure 29: Web application screenshot - Request a new task

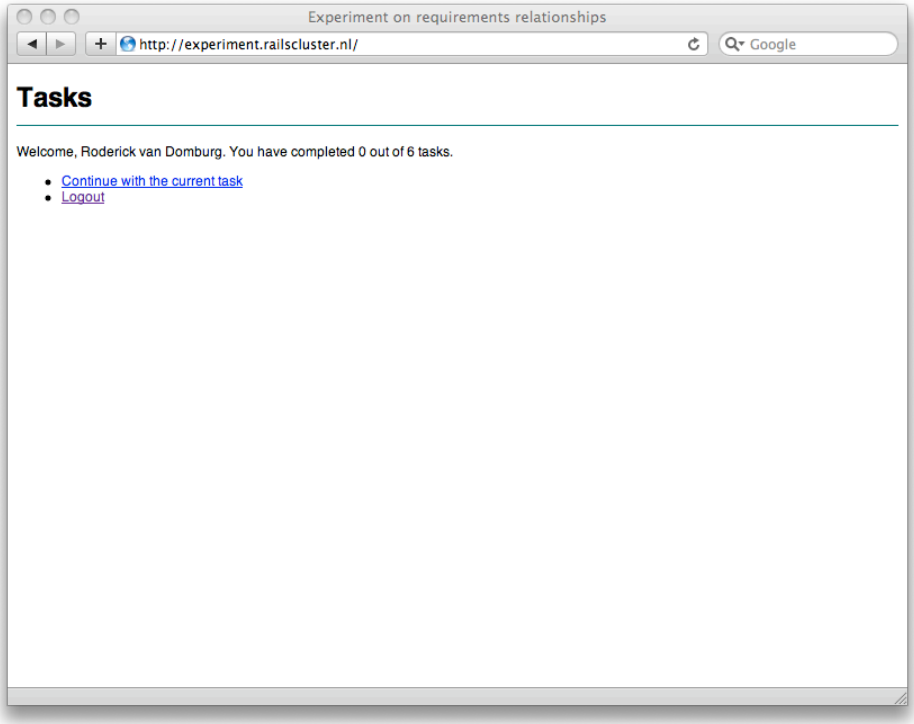


Figure 30: Web application screenshot - Continue the unfinished task

a. Participants request to start a new task. An intermission screen is shown, instructing the user to complete the task once he has started it without taking a break. It contains a button to start a new task. See Figure 31.

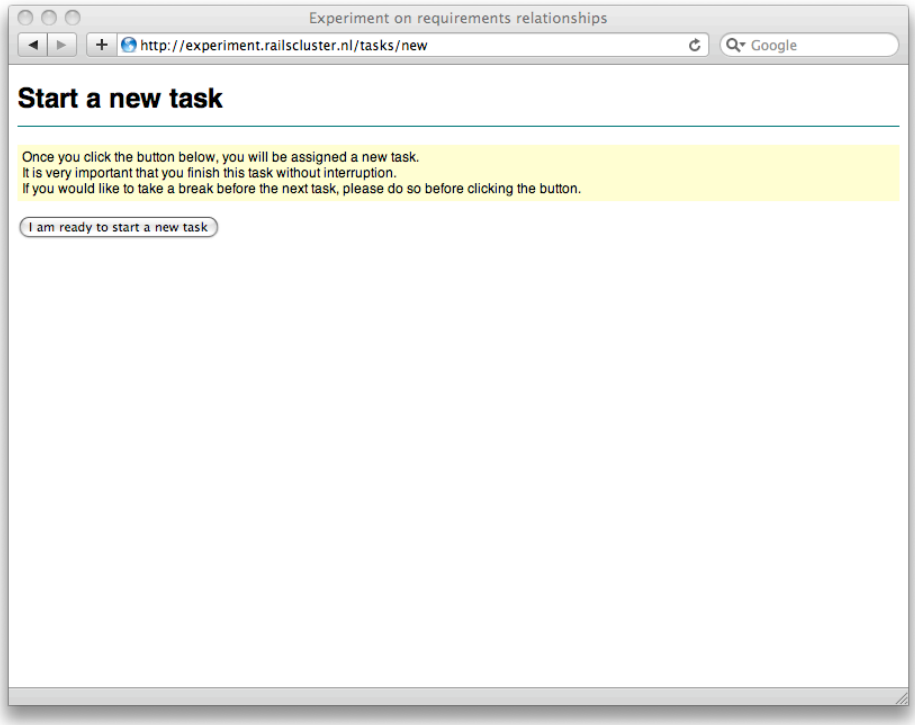


Figure 31: Web application screenshot - Intermission

b. Participant continues the unfinished task. No intermission screen is shown to minimize any loss of time. The participant is shown the task and submission form directly.

2. The first task is performing change impact prediction for a "warming up" task that is the same for all participants. Participants are informed that the result on this warming up task will not count towards their final scores. See Figure 32.

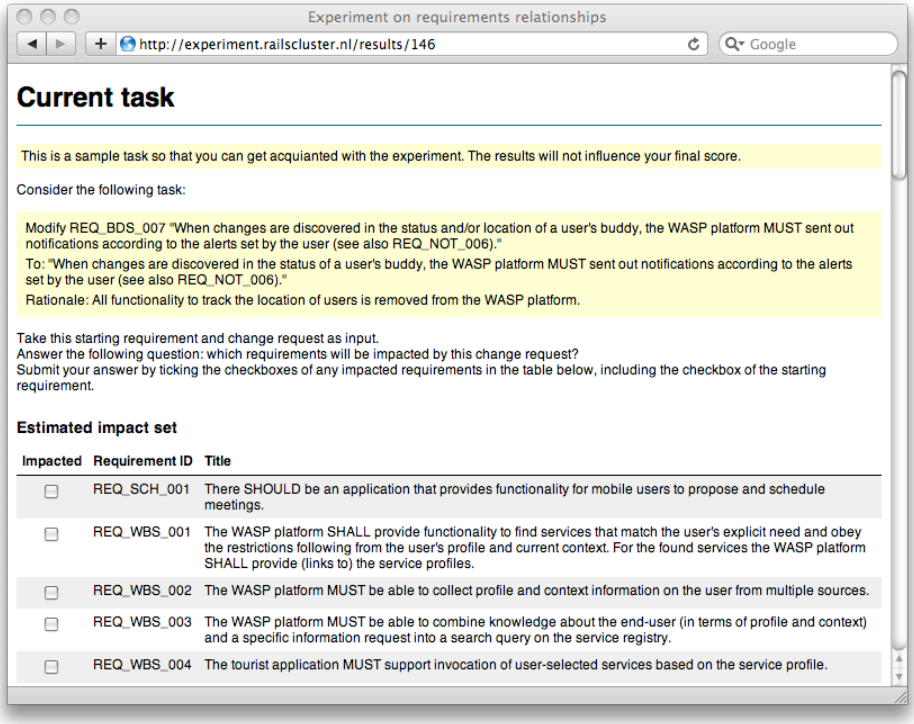


Figure 32: Web application screenshot - Warming up task

3. Pages showing the task contain a description of the task and a list of all requirements with checkboxes that can be ticked if the requirement is deemed to change. A submit button is also present. See Figure 32 and Figure 33.

4. Participants tick impacted requirements and click to submit results. A dialog box pops up asking them if the submission is final since it cannot be changed afterwards. See Figure 33.

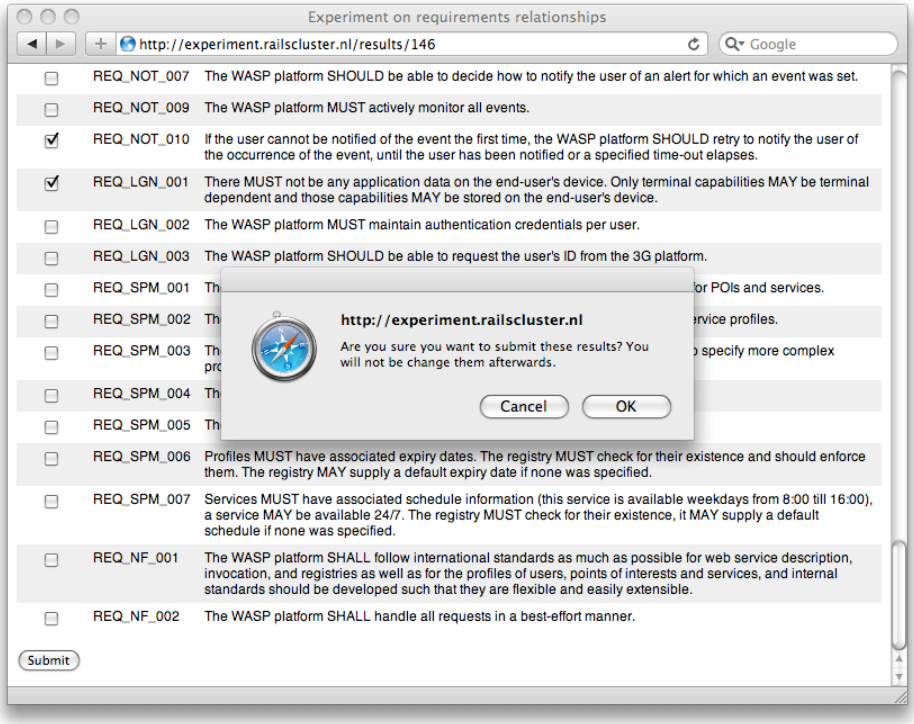


Figure 33: Web application screenshot - Submission

5. Participants submit results. Their Estimated Impact Sets and task times are recorded. They are redirected back to the welcome screen. Participants can now request new tasks, which will be distributed to each participant in random order to rule out learning effects.

6. Once participants have completed all tasks, a welcome message is shown with an instruction to leave the room quietly. See Figure 34.

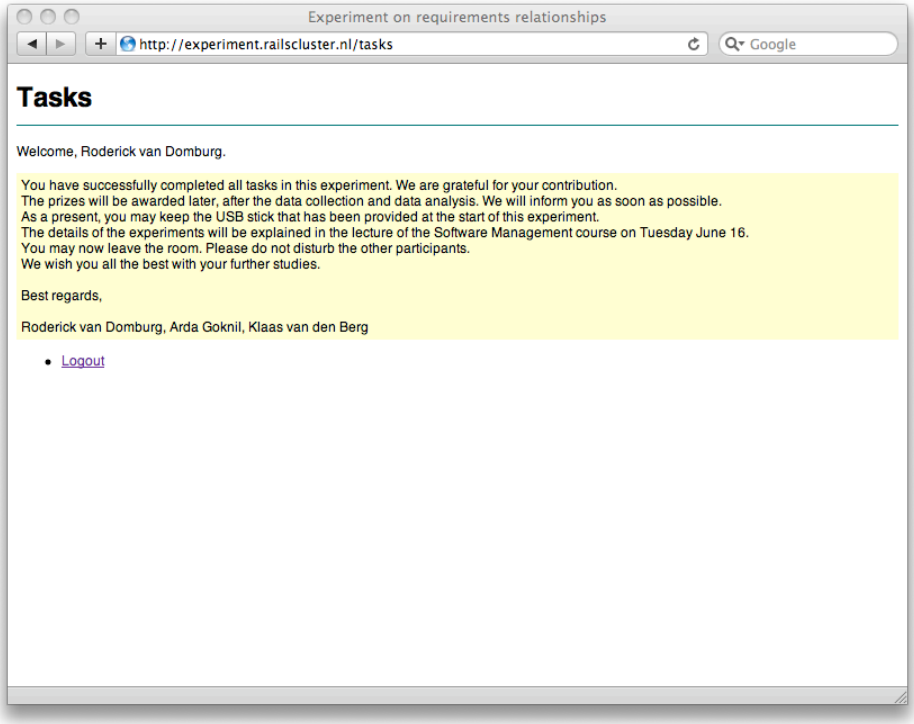


Figure 34: Web application screenshot - Warming up task

The web application is custom-built using the Ruby on Rails framework for web development. It is hosted on an external server but can also be run on a laptop in case of any internet connectivity issues during the experiment. It has 472 lines of code, 693 test lines of code and a code-to-test ratio of 1:1,5 with a line coverage of 74,60%.

Capturing of intermediate products

It was decided not to capture any intermediate products such as the requirements document files from the software tools. The reasoning for this decision can be explained by the Denotation, Demonstration, Interpretation account [36]. See Figure 35.

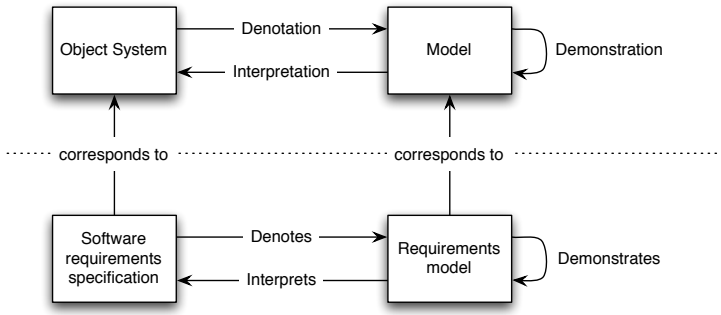


Figure 35: Denotation, Demonstration, Interpretation account [36]

The Denotation, Demonstration, Interpretation account in Figure 35 explains that an object system can be given meaning by denoting it in a model. This model can be used for reasoning about this object system by way of demonstration. Finally, a model can be applied to an object system to interpret it.

In the context of this research, the object systems are software requirements specifications and the models are requirements metamodels. The capturing of intermediate products of change impact prediction implies the capturing of the requirements metamodels that are used as part of the change impact prediction. This research does not attempt to validate the models or metamodels themselves, but rather validate that the models make sense in reality.

Throughout the course of the experiment, the participants are told that they can perform change impact prediction in any way that they see fit using their assigned software tool. In other words: it is the end result that counts, which mimics change impact prediction in the real world. This research is interested in interpretation or the final change impact prediction, not how the software requirements specification was modeled or reasoned on.

From a practical standpoint, the capturing of intermediate products could be interesting for follow-up studies but hard to analyze in the course of this research. There is no theory on change impact propagation with respect to the requirements metamodel, nor are intermediate results comparable between the groups. Finally, none of the software tools support the required recording of user activities.

3.11. Data collection

All covariates, Estimated Impact Sets and task times are collected by the web application. Normally, the Actual Impact Set is determined by actually implementing the change [12]. Because no changes will be implemented in this experiment, the Actual Impact Set is to be determined as a golden standard from experts.

3.12. Analysis procedure

The web application has built-in support to calculate the *F*-scores according to the equation in Chapter 2. For each participant, it will output the participant number, group number, covariate scores and *F*-scores and times per task to a file that can be imported in SPSS 16.0.

SPSS will be used to perform an analysis of variance using planned comparisons to test if participants in the TRIC group had significantly different *F*-scores and times than those in the Microsoft Excel or IBM Rational RequisitePro groups. A similar test will be performed for

analysis of covariance. Finally, a multivariate analysis of variance will be used to test if there are interaction effects between the F-scores and times.

3.13. Validity evaluation

Four types of validity exist [52], each of which is discussed below. Reliability, which is the consistency of the measurement, is a prerequisite for attaining validity, which is the strength of a conclusion, inference or proposition [71].

Statistical conclusion validity

Statistical conclusion validity is defined to be the validity of inferences about the correlation between treatment and outcome: whether the presumed cause and effect covary and how strongly they covary [52].

This research will feature a limited sample set. Statistical power will be low as a result and reliability issues may arise if the domain is poorly sampled [71]. A larger sample of research objects will be required for statistically valid conclusions. The observed power, required sample size for proper power and estimated error will be calculated as part of the analysis.

Internal validity

Internal validity is defined to be the validity of inferences about whether observed covariation between A (the presumed treatment) and B (the presumed outcome) reflects a causal relationship from A to B as those variables were manipulated or measured [52].

First, a strong point of this research is that it studies both tools with and without formal requirements relationship types. However, because the tools are not fully comparable in terms of functionality, maturity and usability, and no statistical adjustments are made, this will still be a threat to external validity and reliability due to uncontrolled idiosyncrasies [71]. Indeed, any inferences will only be valid as they pertain to Microsoft Excel, IBM Rational RequisitePro and TRIC, not to similar applications.

Second, the setup of the lecture is not any fairer by assigning equal slots of time. While an equal amount of time is given to all groups for the lecture, the intelligence and maturity of the tools is very much different. As an example, TRIC and the relationship types will take more time to learn than Microsoft Excel (which is probably already known). By compressing more required knowledge into a shorter timeframe, the intensity of the lecture decreases and participants cannot be expected to understand the software tools equally well.

Using a pre-test and post-test to compensate for learning effects would allow accurately measuring the influence of the instruction on the results [32], although ways to reliably measure aptitude are not directly available and would be a study in itself. An earlier experiment tracked the number of learning hours spent [43] but has not indicated the causality between "number of hours" and "aptitude".

Finally, the lack of theory about what a proper change scenario should be has caused the change scenarios to be developed in a rather ad-hoc fashion, which too hampers reliability due to uncontrolled instrumented variables [71].

Construct validity

Construct validity is defined to be the validity of inferences about the higher order constructs that represent sampling particulars [52].

First, the number of constructs and methods that are used to measure the quality of change impact prediction is fairly monogamous; only the *F*-score is truly a measure of "product" quality. The time taken to complete change impact predictions is more of a measure of "process" quality. This may underrepresent the construct of interest, complicate inferences and mix measurements of the construct with measurement of the method [52].

Second, the validity of constructs is further threatened by reactivity to the experimental situation, also known as Hawthorne effects [52], which is also a concern for reliability of individuals [71].

External validity

External validity is the validity of inferences about whether the cause-effect relationship holds over variation in persons, settings, treatment variables and measurement variables [52].

First, not only is the low sample size a threat, but so is the fact that there is only a single software requirements specification as research object. As with the internal validity of the software tools, any inferences will only be valid as they pertain to the WASP requirements specification.

Second, the research participants may not represent real-world software maintenance engineers and the lecturers are three different people, which poses more threats to external validity [52] and is concern for reliability in instrumented variables [71].

3.14. Conclusion

A blueprint experimental design was developed in a goal-oriented fashion. It is hypothesized that using TRIC, a software tool with formal requirements relationship types, will positively impact the quality of change impact predictions.

This hypothesis was found testable with a quasi-experiment using a synthetic design, by giving different groups of participants the same change scenarios and software requirements specification to perform change impact prediction on, yet assigning them with different software tools. A process that is highly comparable between the groups and web application were developed to administer the experiment.

An evaluation of validity found a low level of external validity, which is acceptable considering the intended contribution to provide a blueprint for future experiments.

The internal validity seems strong as long as the inferences pertain to the three software tools being used, as opposed to their classes of tools, but is hampered by an unfair lecturing setup and lack of theory surrounding change scenarios. The validity can therefore only be regarded as mediocre in spite of best effort.

4. Execution

4.1. Introduction

This chapter describes each step in the production of the research [73]. It describes the practical steps taken to execute the experiment, including the participating people and how they were randomized over the groups, the setup of the environment and instrumentation and how data was collected. Finally, any recorded deviations in validity during the execution are noted.

4.2. Sample

The experiment was conducted with 22 participants. 21 of these participants completed the online registration before the start of the experiment to score the covariates and facilitate group matching. 2 participants did not pre-register. Their responses to the registration were added after the execution of the experiment. All participants who registered also showed up.

The final distribution of the participants over the group is shown in Table 11.

Group	Participants
Microsoft Excel	6
IBM Rational RequisitePro	7
TRIC	8

Table 11: Participant distribution over groups

This distribution was determined by:

1. The pre-experiment participant registration and group matching
2. One dropout in the IBM Rational RequisitePro group
3. The pragmatic assignment of latecomers

These factors are detailed later in this chapter.

4.3. Preparation

The following practical preparations were conducted:

- **Room reservations.** Three locations were booked with the facility management of the University of Twente a month in advance; one location per group. Two of the three assigned locations were computer clusters in a single large room with a total of four clusters. The third location was a computer cluster in a room on the first floor. The rooms were not comparable in terms of environment or layout. No three neutral rooms were available.

  It was decided to place the group with Microsoft Excel support in the room on the first floor and the other two groups in the single large room. The single large room was also determined to be the meeting place for all participants. This configuration maximized the available time and supervisor support for installing IBM RequisitePro and TRIC. See the following bullet point.

  Unfortunately there was loud construction work ongoing in the building in the vicinity of the large room. This hampered the lecturing and probably participant focus. The room was also used by other students who were quite noisy, adding up to the same effect. This situational factor is a reliability problem [71].

- **Software installation.** The computers in all three locations had Microsoft Excel 2003 installed. Because of IT management policy and time constraints, it was not possible to preinstall IBM RequisitePro or TRIC on the computers when requested two days before the experiment. As a workaround, the following files were copied onto the USB memory sticks for the participants:

  - The setup file and a trial license key for IBM Rational RequisitePro
  - The setup file for TRIC
  - Self-written README files with instructions for installing the above
  - The PDF version of the WASP case
  - A URL file to the online web application

  Two days before the experiment, all participants were asked to indicate if they had a laptop running Microsoft Windows (a requirement for running IBM Rational RequisitePro) and if they would bring it.

  The IBM RequisitePro and TRIC groups were instructed to install their respective software tools with supervisor assistance. The time taken for setup was not measured or deducted from any other experiment activities.

One participant that ran a Chinese version of Microsoft Windows and was part of the IBM RequisitePro group was unable to install that software tool. Consequently he was removed from the experiment execution.

- **Ordering of drinks.** An order was placed with the catering service of the University of Twente one week in advance for one soft drink and one glass of water for 30 participants. It did not arrive in cans, as was ordered, but in bottles and was thus more of a hassle during the breaks than anticipated.

- **Beamer reservations.** Three beamers and two projector screens were booked with the facility management of the Faculty of Electrical Engineering, Mathematics and Computer Science one week in advance. A third project screen was unavailable. When collecting the projector screens, only one had been prepared by the facility management.

  The beamers were setup and tested in the experiment locations three hours before the start of the experiment. The projector screen was placed in the room with the single computer cluster on the first floor, which had no other opportunities for projecting the beamer on. The unequal equipment may be a reliability problem [71] although this is unlikely; the beamers in the single large room projected against the wall and this was deemed legible by the participants.

- **Lecture preparation.** Five slideshows were created: one for the general instruction, three for the specific instruction (one per group) and one for the general kick-off. These were created by supervisor Arda Goknil, discussed in two meetings and subsequently tuned. It was agreed only to lecture what is on the sheets and not to answer any questions regarding the content.

- **Participant distribution.** The participants were distributed over three groups. The randomized, tuned and final participant distribution is available in Appendix C. The final distribution is different from the tuned distribution because latecomers were assigned to a group which had not begun the lecture yet, one participant dropped out because he could not install IBM Rational RequisitePro on his Chinese Microsoft Windows. See Table 12.

Distribution	Participants			Distance
	Microsoft Excel	IBM Rational RequisitePro	TRIC	
Randomized	7	6	7	32
Tuned	7	6	7	18
Final	6	7	8	20

Table 12: Distances between the participant distributions

Table 12 summarizes the number of participants per group per distribution. It also shows the distance, as calculated as the sum of the differences in covariate scores between all pairs of groups. A distance of 0 would indicate no differences in covariate scores.

4.4. Data collection performed

All 22 participants submitted estimated impact sets for six change scenarios. Consequently 132 estimated impact sets were collected. Of these, 22 were the result of warm-up scenarios and were not used in statistical analysis.

4.5. Validity procedure

There were some deviations from the planning with regards to the experiment location (noisy environment, lack of equipment and a faulty delivery of soft drinks) and participant distribution (latecomers and drop-outs). These were discussed above.

Supervisors surveyed the participants from walking distance throughout the course of the experiment. They noted the following deviations:

- **Lack of focus.** Not all students were as focused on the task as expected, in spite of the monetary rewards offered. One student was actively listening to music and seen watching YouTube videos during the experiment. Nothing was done about this, because it is uncertain if such behavior is representative or not. This matches observations regarding discipline in experiments with design recording [1] and may be a reliability problem [71].

- **Ambiguous rationales.** As discussed in the Chapter 3, the change scenarios are not entirely unambiguous. Some students raised questions about the rationale. As with the lectures, the supervisors withheld themselves from providing further explanation. This may be a reliability problem because it can induce guessing with individuals [71].

- **Lack of time.** Many students were not finished with adding relationships before the break. After the break, some of them tried catching up by adding more relationships. Others started change impact prediction with the unfinished set of relationships. When this was noticed, the supervisors jointly decided to provide an extra 15 minutes. The extra time was not enough for many students. This situational factor may be a reliability problem [71].

- **Ineffective use of tools.** Not all students used the tool to full effect and some did not use them at all. Nothing was about this, because the participants were told to use the software tool and documents in any way they saw fit. This may be a reliability problem due to differences in skills and ability if not corrected for by covariates [71].

- **Lack of precision.** Some participants did not check the initially changed requirement as part of their Estimated Impact Set, even though they were instructed to do so both during the lecture and by the web application. The data set was corrected to include the initially changed requirement for all participants. The underlying assumption is that this has been an oversight by the participants, however, it may just as well be a reliability problem due to a lack of motivation, concentration or reading ability [71].

4.6. Conclusion

The execution generally proceeded well and as planned but suffered from a poor and unequal environment. An evaluation of validity revealed that the participants were under time pressure to complete the experiment and that some had a lack of focus, precision and effectiveness in using the assigned tool, which are concerns for reliability.

5. Analysis

5.1. Introduction

This chapter summarizes the data collected and the treatment of data [73]. A number of analyses were made regarding the representativeness of the change scenarios, the inter-rater reliability of the golden standards and finally the quality of participants' change impact predictions and the time they took to complete them.

5.2. Change scenario representativeness

Due to time constraints, the change scenarios that were created were discussed with an expert only after the experiment was conducted. The expert was one of the original authors of the WASP specification. He is still working at the Telematica Instituut, now renamed to Novay, where the WASP specification has been produced.

The expert was consulted by way of a face-to-face interview at Novay. The central question in the interview was: can you provide us with your golden standard based on these change scenarios? Because of time constraints he was unable to do so, although he did have time to explain the background of the WASP specification and check the representativeness of the change scenarios.

The WASP specification does not mention a goal, although one was revealed in an interview with an author of the WASP specification: to shift mobile services from an service provider-centric perspective to an independent network. This goal should also have formed the backdrop of real-world change scenarios [34]. Experts indicate that the omission of the goal reduces the clarity of the document, see Appendix A.

On an ordinal scale of low, medium to high, the expert from Novay rated the representativeness of the change scenarios as follows. The task numbers correspond to the change scenario numbers. See Table 13.

Scenario	Representativeness	Comment
1	Medium	Possible. In the light of the goal, it would have been better to remove "the 3G platform" for better independency and flexibility.
2	Low	Not likely to happen because the service profiles are at the heart of the system.
3	High	This is precisely what is happening right now. Additionally, services could be combined to provide higher accuracy.
4	Medium	Possible, but not to the informal reason behind the requirement, which assumes that there already is calculated route such as a walking tour.
5	Low	Not likely to happen because context-awareness is at the heart of the system.
Warming up	Low	Not likely to happen because location-awareness is at the heart of the system. Better would be to add limitations based on privacy directives.

Table 13: Representativeness of change scenarios as rated by Novay expert.

5.3. Golden standard reliability

The establishment of a golden standard was initiated after the experiment was conducted. Four people created a golden standard individually; one expert (another original author from the WASP specification still with Novay) and three academics with the software engineering department and the QuadREAD Project: a postdoc, a PhD candidate and a master student.

The golden standards contain dichotomous data: a requirement is rated to be either impacted or not impacted. In Appendix D, these ratings are coded as "1" (impacted) and "0" (not impacted).

To create the final golden standard, it was decided to use the mode of the individual golden standards. When this was not possible initially because of a split, then the academics debated until one of them was willing to revise his prediction. Revised predictions are indicated with an asterisk in Appendix D.

Inter-rater reliability

In an experimental setting, it is important to calculate the level of agreement between expert ratings [30] such as the golden standards. This is called the inter-rater reliability.

The calculation of the inter-rater reliability depends on the type of data that has been rated and if there are two raters or multiple raters. This experiment features four raters who have independently produced a golden standard for dichotomous data. A recommended method for calculating the inter-rater reliability for dichotomous data with multiple raters is to calculate the raw agreement and intraclass correlation [60].

Raw agreement indices

Raw agreement indices have a unique common-sense value and are important descriptive statistics. They are informative at a practical level [60].

The overall proportion of observed agreement is calculated by dividing the total number of actual agreements by the number of possible agreements [60]. An agreement is a binary relationship between two raters that have rated a case (requirement) in the same category (impacted or not).

A nonparametric bootstrap can be used to estimate the standard error of the overall agreement. This can be performed using a nonparametric test for several related samples under the assumption that the cases are independent and identically distributed [60]. This assumption can be accepted because the same raters rate each case and there are no missing ratings.

Significance testing

A test for significance can be used to analyze if the golden standards have any significant differences between them [49]. A plausible test for significance is the Friedman Test, which tests the null hypothesis that measures (ratings) from a set of dependent samples (cases) come from the same population (raters). The Friedman Test is asymptotic and therefore does not provide exact significance levels [40].

A nonparametric bootstrap can be used for significance testing at greater confidence levels. While the use of the Monte Carlo approach is suggested [60], exact tests are more advantageous for this research. Unlike Monte Carlo estimates, exact tests do not overfit the data and are more precise at the cost of being more computationally expensive [23].

Intraclass correlation classes

The intraclass correlation assesses rating reliability by comparing the variability of different ratings of the same subject to the total variation across all ratings and all subjects [60]. Three classes of intraclass correlation for reliability can be identified, named Case 1, Case 2 and Case 3 [53]. See Table 14.

Case	Design
1	Raters for each subject are selected at random.
2	The same raters rate each case. These are a random sample.
3	The same raters rate each case. These are the only raters.

Table 14: Different types of intraclass correlation [53]

In this research, the same people rate all cases; the golden standards for each scenario are made by the same raters. Case 2 and Case 3 would both apply: on the one hand the raters could be said to be randomly picked from a greater population of experts and academics. This is supported by the knowledge that it was attempted to find more experts and academics to create golden standards. Case 2 would apply.

It could also be debated that the current experts and academics are not necessarily representative for their kind. Case 3 would then apply. With arguments to support both, Case 2 is selected because Case 3 does not allow for generalizations and is thus used infrequently [60].

For all three cases, there are two methods that can be used for calculating inter-rater reliability: consistency and absolute agreement. Consistency should be chosen if the relative standing of scores is important; absolute agreement if the scores themselves also matter [4]. The latter applies to this research, so the absolute agreement method is chosen.

Results

The results of the analyses for raw agreement, significance and intraclass correlation is shown in Table 15. While the interpretation is provided in the Chapter 6, a guideline for reading these results is provided here:

- Significance levels equal to or less than 0,0005 indicate that there were significant differences between the golden standards. Exact significance levels provide more precise values than asymptotic significance levels. Asymptotic significance levels are provided for comparison with other experiments that do not list exact significance levels.

- The intraclass correlation score indicates the level of agreement. Higher scores are better, with a score of "0" indicating no agreement and a score of "1" indicating full agreement. Further criteria for classifying the level of agreement based on intraclass correlation score are provided in Chapter 6.

Golden standard for task	Impacted set size	Raw agreement		Significance [a]		Intraclass correlation
		Mean	Standard error	Asymptotic	Exact	Two-way random [b]
1 - Initial	-	51,0%	6,5%	0,038	0,036	0,760
1 - Revised	3	58,1%	9,1%	0,343	0,519	0,832
2 - Initial	-	71,4%	4,5%	0,709	0,823	0,909
2 - Revised	9	78,6%	4,2%	0,438	0,544	0,936
3	1	100,0%	0,0%	-	1,000	1,000
4	1	100,0%	0,0%	-	1,000	1,000
5	6	44,9%	9,7%	0,000 [c]	0,000 [c]	0,712

a. Friedman Test

b. Using an absolute agreement definition between four raters

c. p<0,0005

Table 15: Inter-rater reliability analysis

5.4. Precision-Recall and ROC graphs

Precision-Recall and ROC graphs can present the results of performing the change impact predictions graphically. Box plots of the *F*-scores and times are available in Appendix E. Precision-Recall and ROC graphs are available in Appendix F.

5.5. One-way between-groups ANOVA

One-way between-groups analysis of variance is used when there is one independent variable with three or more levels and one dependent continuous variable. It tests if there are significant differences in the mean scores on the dependent variables, across the three groups [49].

Following Hypothesis 1 and Hypothesis 2, the analysis should test if the TRIC group performed superior to both the Microsoft Excel and IBM Rational RequisitePro groups. Planned comparisons lend themselves better to this goal than post-hoc tests because of power issues. Planned comparisons are more sensitive in detecting differences, because post-hoc tests set more stringent significance levels to reduce the risk of false positives given the larger number of tests that are performed [49].

In this experiment, the independent variable is the experiment group. This experiment features two dependent variables, the *F*-score and the time of a task, and an analysis of variance can be performed separately on both.

A number of assumptions underlie analyses of variance. These assumptions must be tested for the actual analyses to be carried out [49]. There were some deviations while testing for normality and homogeneity of variance, which are discussed below.

Testing for normality

One assumption of analyses of variance is that the data is normally distributed. This can be done by assessing histograms and normal probability plots for both the *F*-score and time per task per group. Alternatively, the Kolmogorov-Smirnov statistic may be calculated to assess the normality of the distribution of scores [49].

An assessment of histograms and normal probability plots was inconclusive. Due to the low sample size, the number of scatter points was too low to conclude either normality or non-normality.

As an alternative, Table 16 shows the Kolmogorov-Smirnov statistic for assessing normality. Here, Kolmogorov-Smirnov scores equal to or less than 0,05 indicate non-normality.

Task	Group	Kolmogorov-Smirnov	
		F-score	Time
1	Excel	0,125	0,200
	RequisitePro	0,141	0,200
	TRIC	0,200	0,200
2	Excel	0,200	0,200
	RequisitePro	**0,049**	0,200
	TRIC	0,101	**0,023**
3	Excel	0,200	0,173
	RequisitePro	0,200	0,200
	TRIC	0,124	0,113
4	Excel	0,200	0,200
	RequisitePro	0,200	0,185
	TRIC	**0,045**	**0,027**
5	Excel	0,200	0,094
	RequisitePro	**0,000**	0,200
	TRIC	**0,003**	0,200

Table 16: Kolmogorov-Smirnov tests of normality

Table 16 shows six cells in boldface for which the Kolmogorov-Smirnov test is significant at p<0,05. These cells thus violate the assumption of normality. Box plots for all results are available in appendix E.

Fitting the population

In social sciences, data that is generated by experiments is not normally distributed, especially in the case of large sample sizes [49]. A standard procedure is to remove outliers to make the population fit. That approach is problematic here because of the following reasons:

- The number of cases is already small. First, removing cases decreases statistical power further. Second, the low amount of cases may in fact be the reason for non-normality.

- For those results that are not normally distributed, the mean and 5% trimmed mean (table omitted for brevity) are quite similar. Normally, differences between the mean and 5% trimmed mean indicate the presence of outliers. The fact that some results are not normally distributed, yet have comparable means and 5% trimmed means, indicates that there are many extreme scores on either side of the mean. This supports the notion of non-normality and leads to the conclusion that the data set cannot be fitted to a normal distribution.

Consequently, the result set was not fitted to a normal distribution.

Consequences

Data sets which contain non-normally distributed results may not be assessed using an analysis of variance, but can be assessed using a non-parametric test that does not make any assumptions about the distribution of data. The downside to this approach is that non-parametric tests are less sensitive to detecting differences. Analyses of variance are preferable over non-parametric tests if the data set allows [49].

The analysis of variance will only address tasks 1 and 3 for which normality can be assumed. Following that, a non-parametric test will be performed with tasks 2, 4 and 5.

Homogeneity of variance testing

Another assumption of analyses of variance is that the variance in responses to independent variables is approximately the same in all groups [49]. This is not the case with task 3, which violates the homogeneity of variance assumption using Levene's test at p=0,021 (table omitted for brevity) at p<0,05. This is not problematic to the analysis of variance however, because the size of the groups is reasonably similar;

$\frac{\text{largest}}{\text{smallest}} = \frac{8}{6} = 1,33$, while the maximum tolerable value is 1,5 [56].

Consequently, homogeneity of variance may be assumed for all results.

Results for F-score

Table 17 presents the results of a one-way between-groups analysis of variance to explore the impact of using three different software tools on the quality of change impact predictions, as measured by the *F*-score. A planned comparison is used to compare the TRIC group to the Excel and RequisitePro groups.

Only tasks 1 and 3 met the preconditions for performing an analysis of variance; tasks 2, 4 and 5 are tested using a non-parametric test in a paragraph 5.6.

While the interpretation is provided in Chapter 6, a guideline for reading these results is provided here:

- Significance levels equal to or less than 0,005 indicate a significant difference in *F*-scores between the TRIC group and the other two groups.
- The "F" column lists the test statistic for an analysis of variance with a F-distribution and should not be confused with the *F*-score of a change prediction. It is used to describe the shape of the distribution of the analysis of variance. It is reported for comparison with other experiments.
- The η^2 value describes the ratio of variance explained in the *F*-score by the group assignment. By multiplying it with 100, it can be interpreted as the percentage that the group assignment had on the variance in *F*-scores. Like the statistic for the F-test, it is also useful for comparison with other experiments.

Task	Group	*F*-score (higher is better)		One-way ANOVA [a]		
		Mean	Standard deviation	Significance	F	η^2
1	Excel	0,498	0,232	0,866	0,030	0,002
	RequisitePro	0,658	0,187			
	TRIC	0,593	0,176			
	Total	0,588	0,198			
3	Excel	0,407	0,321	0,629	0,242	0,013
	RequisitePro	0,468	0,290			
	TRIC	0,507	0,325			
	Total	0,465	0,300			

a. Using a planned comparison with TRIC

Table 17: One-way between-groups ANOVA on F-score

Using a planned comparison for the TRIC group, there were no statistically significant differences at the p<0,05 level in the *F*-scores of the three groups in either task 1 [F(1, 18)=0,030; p=0,866] or task 3 [F(1, 18)=0,242; p=0,629].

Results for time

Table 18 presents the results of a one-way between-groups analysis of variance to explore the impact of using three different software tools on the time taken to complete change impact predictions as measured in seconds.

Only tasks 1 and 3 met the preconditions for performing an analysis of variance; tasks 2, 4 and 5 are tested using a non-parametric test in paragraph 5.6.

The guidelines for reading the results for *F*-scores apply here similarly.

Task	Group	Time (lower is better)		One-way ANOVA [a]		
		Mean	Standard deviation	Significance	F	η^2
1	Excel	193	89	0,000	24,04	0,572
	RequisitePro	137	53			
	TRIC	368	117			
	Total	241	136			
3	Excel	172	70	0,219	1,753	0,088
	RequisitePro	239	121			
	TRIC	314	219			
	Total	249	161			

a. Using a planned comparison with TRIC

Table 18: One-way between-groups ANOVA on time

There was a statistically significant difference at the p<0,05 level in the time of the three groups for task 1 [F(1, 18)=24,04; p=0,000]. The effect size, calculated using η^2, was 0,572. In Cohen's terms, the difference in mean scores between the groups is large [13]. The TRIC group performs change impact predictions 48% slower than the Microsoft Excel group and 63% slower than the IBM Rational RequisitePro group.

There was no statistically significant difference at the p<0,05 level in the times of the three groups for task 3 [F(1, 18)=1,753; p=0,219].

Statistical power

The attained statistical power is *56%* for detecting effects with a large size, p<0,05; sample size 21 and 18 degrees of freedom. The critical value for the F-test statistic to attain a significant result is 4,41.

To attain a statistical power of 80% a sample size of 34 would be required. The critical value for the F-test statistic to attain a significant result would be 4,15. This was calculated using the G*Power 3 software tool [13] because SPSS 16.0 lacked the necessary support.

5.6. Non-parametric testing

As a non-parametric test, χ^2 test for goodness of fit can test if there are significant differences between dependent variables across multiple groups without requiring a normal data distribution [49]. It does require a sufficiently large sample size; values of 20 through 50 have been reported although there is no generally agreed threshold [23].

Table 19 and Table 20 display the results of a χ^2 test for tasks 2, 4 and 5, which did not meet the requirements for analyzing them using a more sensitive analysis of variance.

Results for F-scores

Table 19 presents the results of a χ^2 test to explore the impact of using three different software tools on the quality of change impact predictions, as measured by the *F*-score.

Tasks 2, 3 and 5 did not meet the preconditions for performing the preferred analysis of variance; tasks 1 and 3 are tested using an analysis of variance in paragraph 5.5.

While the interpretation is provided in Chapter 6, a guideline for reading these results is provided here:

- Significance levels equal to or less than 0,005 indicate a significant difference in *F*-scores between the TRIC group and the other two groups.
- The χ^2 value describes the test statistic for a χ^2 test. It is used to describe the shape of the distribution of the χ^2 test. It is reported for comparison with other experiments.

Task	Group	*F*-score (higher is better)		χ^2	
		Mean	Standard deviation	Significance	χ^2
2	Excel	0,499	0,319	0,584	1,077
	RequisitePro	0,517	0,129		
	TRIC	0,424	0,275		
	Total	0,476	0,242		
4	Excel	0,407	0,182	0,717	0,667
	RequisitePro	0,524	0,230		
	TRIC	0,461	0,161		
	Total	0,467	0,188		
5	Excel	0,423	0,160	0,444	1,625
	RequisitePro	0,528	0,100		
	TRIC	0,573	0,151		
	Total	0,515	0,146		

Table 19: χ^2 test for goodness of fit on F*-score*

There were no statistically significant differences at the p<0,05 level in the *F*-scores of the three groups in task 2 [χ^2=1,077; df=2; p=0,584], task 4 [χ^2=0,667; df=2; p=0,717] or task 5 [χ^2=1,625; df=2; p=0,444].

Time

Table 20 presents the results of a χ^2 test to explore the impact of using three different software tools on the time to complete change impact predictions, as measured in seconds.

Tasks 2, 3 and 5 did not meet the preconditions for performing the preferred analysis of variance; tasks 1 and 3 are tested using an analysis of variance in paragraph 5.5.

The guidelines for reading the results for *F*-scores apply similarly.

Task	Group	Time (lower is better)		χ^2	
		Mean	Standard deviation	Significance	χ^2
2	Excel	133	83	0,000	414
	RequisitePro	154	76		
	TRIC	222	137		
	Total	174	107		
4	Excel	213	111	0,000	102
	RequisitePro	300	81		
	TRIC	467	248		
	Total	339	196		
5	Excel	324	274	0,000	612
	RequisitePro	170	64		
	TRIC	342	133		
	Total	280	181		

Table 20: χ^2 test for goodness of fit on time

There were statistically significant differences at the p<0,05 level in the times of the three groups in task 2 [χ^2=414; df=2; p=0,000], task 4 [χ^2=102; df=2; p=0,000] and task 5 [χ^2=612; df=2; p=0,000].

Post-hoc comparison

Because χ^2 tests do not support planned comparisons, a post-hoc comparison is required to discover how groups differ from each other. Post-hoc comparisons explore the differences between each of the groups and can be performed using a Mann-Whitney U test, which tests for differences between two independent groups on a continuous measure [49].

A post-hoc comparison using a Mann-Whitney U test revealed that the time taken to complete task 4 was significantly different between the Microsoft Excel and TRIC groups, p=0,020. The TRIC group performs change impact predictions 54% slower than the Microsoft Excel group.

A similar post-hoc comparison revealed that the time taken to complete task 5 was significantly different between the IBM Rational RequisitePro and TRIC groups, p=0,011. The TRIC group performs change impact predictions 50% slower than the IBM Rational RequisitePro group.

No other combination of groups yielded a significant difference in time results in the post-hoc test. This includes task 2, even though it was indicated to be significantly different by the initial χ^2 test.

Statistical power

The attained statistical power for the χ^2 tests is 52% for detecting effects with a large size, p<0,05, sample size 21 and two degrees of freedom. The critical χ^2 value to attain a significant result is 5,99.

To attain a statistical power of 80% a sample size of 39 would be required. The critical χ^2 value to attain a significant result would remain 5,99. This was calculated using G*Power 3 [13] because SPSS 16.0 lacked the necessary support.

5.7. Analysis of covariance

Analysis of covariance is an extension of analysis of variance that explores differences between groups while statistically controlling for covariates [49]. As an extension of analysis of variance, it can only be used for tasks 1 and 3 for which the initial assumptions were met.

Analyses of covariance require additional assumption testing. There was a deviation in the reliability testing of covariates, which inhibited an analysis of covariance to be conducted.

Reliability of covariates

The set of covariates should be sufficiently reliable to perform an analysis of covariance. Cronbach's alpha is an indicator of internal consistency and can be used to measure this reliability. A sufficient level of reliability as measured by Cronbach's alpha is 0,7 or above [49]. However, Cronbach's alpha for the covariates in this experiment is only 0,310 which indicates poor reliability.

Following this initial result, several covariates were removed in an attempt to attain sufficient reliability. The set of covariates with the highest reliability consists of the following, from the most to the least contributing:

- Completion of a basic requirements engineering course
- Completion of an advanced requirements engineering course
- Requirements management experience
- Gender

The covariates which were detrimental to the reliability were, from the most to the least detrimental:

- Current educational program
- Nationality
- Level of formal education

The reliability of this reduced set of covariates is 0,585. Although it is an improvement over the initial reliability, it is still too low to warrant a reliable analysis of covariance.

5.8. Multivariate analysis of variance

Multivariate analysis of variance is an extension of analysis of variance when there is more than one dependent variable such as is the case with the *F*-score and time. The advantage of performing multivariate analyses of variance over performing separate one-way analyses of variance is that the risk of false positives is reduced [49].

Analyses of covariance require additional assumption testing. There was a deviation in the reliability testing of covariates, which inhibited an analysis of covariance to be conducted.

Linearity

The multivariate analysis of variance assumes that the *F*-score and time are related in some way. Statistically, there should be a straight-line relationship between them [49].

The linearity of two variables may be assessed by inspecting a scatterplot of *F*-score versus time per task per group. Alternatively, a Pearson product-moment correlation calculation may be performed, which calculates the linearity between two variables [49].

An assessment of the scatterplots was inconclusive. Due to the low sample size, the number of scatter points was too low to conclude either normality or non-normality.

Table 21 displays the results of a Pearson product-moment correlation calculation. Significance levels equal to or less than 0,005 indicate linearity between the *F*-score and time.

Task	Group	Correlation	
		Significance	Effect size
1	Excel	0,911	0,059
	RequisitePro	0,879	0,071
	TRIC	0,751	-0,134
3	Excel	0,559	0,303
	RequisitePro	0,643	-0,215
	TRIC	0,807	-0,104

Table 21: Pearson product-moment correlation for F*-score and time*

There are no significant correlations at the $p<0,05$ level for either task 1 or task 3.

Transformation strategies using either a square root or logarithmic function can be used in an attempt to attain linearity over a skewed data set [48]. Both were attempted but yielded no significant results.

5.9. Conclusion

The results for tasks 1 and 3 could be analyzed using the planned analysis procedure, but the results for tasks 2, 4 and 5 were not normally distributed and had to be tested with a non-parametric χ^2 test. Tests for significance revealed no significant differences for any of the groups and tasks on *F*-score. Significant differences between the groups in time were discovered for tasks 1, 4 and 5.

Planned covariate and multivariate analyses of variance could not be executed, respectively due to reliability and linearity issues.

6. Interpretation

6.1. Introduction

This chapter interprets the findings from the analysis presented in Chapter 5 [73]. It retains the paragraph structure from Chapter 5 to improve readability.

6.2. Change scenario representativeness

Not all change scenarios were judged to be representative. This is both a reliability problem and a threat to internal validity: this research attempts to reflect the real world yet does not fully have real-world change scenarios.

As the next paragraph will turn out, the golden standards are very reliable. This can only be true if the change scenarios have a low level of ambiguity. This partly offsets the low representativeness: although the change scenarios may not reflect the real world, they can still be well-understood and applied to the WASP specification.

6.3. Golden standard reliability

Criteria exist to classify intraclass correlation scores [22]. See Table 22.

Score	Classification
< 0,4	Poor
0,4 - 0,59	Fair
0,6 - 0,74	Good
> 0,74	Excellent

Table 22: Intraclass correlation score classifications [22]

Using this classification for the intraclass correlation and the raw agreement and significance listed in Table 22, the following interpretation can be provided.

Task 1

The initial golden standards for task 1 have a raw agreement score of 51,0%. The results of both tests for significance do not suggest any significant differences between the golden stan-

dards, p>0,0005. The intraclass correlation score of 0,760 suggests excellent inter-rater reliability.

The revised golden standards for task 1 have a raw agreement score of 58,1% which constitutes an improvement of 7,1 percentage points. The results of both tests for significance do not suggest any significant differences between the golden standards, p>0,0005. The intraclass correlation score of 0,832 suggest excellent inter-rater reliability.

Task 2

The initial golden standards for task 2 have a raw agreement score of 71,4%. The results of both tests for significance do not suggest any significant differences between the golden standards, p>0,005. The intraclass correlation score of 0,909 suggests excellent inter-rater reliability.

The revised golden standards for task 2 have a raw agreement score of 78,6% which constitutes an improvement of 7,2 percentage points. The results of both tests for significance do not suggest any significant differences between the golden standards, p>0,0005. The intraclass correlation score of 0,936 suggest excellent inter-rater reliability.

Task 3

The golden standards for task 3 have a raw agreement score of 100,0%; all raters were in full agreement. The asymptotic significance of the Friedman Test could not be calculated because the balance between the raters exceeds 90:10, which is a criterium for the Friedman Test [49]. The result of the exact test of significance does not suggest any differences between the golden standards at all, p=1,000. The intraclass correlation score of 1,000 suggests perfect inter-rater reliability.

Task 4

The golden standards for task 4 score equally to the golden standards of task 3.

Task 5

The golden standards for task 5 have a raw agreement score of 44,9%. The results for both tests of significance do suggest significant differences between the golden standards, p<0,0005. The more precise intraclass correlation score of 0,712 does suggest good inter-rater reliability.

Overall

The inter-rater reliability of the golden standards is very high; two are perfect, two are excellent and one is good. Consequently, the golden standard which served as Actual Impact Set is very reliable.

The high inter-rater reliability also means that the design of the tasks is feasible. Had they been too ambiguous, then it would have been likely that the inter-rater reliability would have been much lower. While it is still uncertain how change scenarios influence the end results, these results support our research design.

6.4. Precision-Recall and ROC graphs

Like the raw agreement indices, the Precision-Recall and ROC graphs are usually presented because they offer a common-sense value to readers. The graphs of the results from this data set do not offer such value because there are no clear differences. This is supported by the findings from the analyses of variance.

6.5. One-way between-groups ANOVA

Because of violations during assumption testing, only tasks 1 and 3 could be tested using an analysis of variance, which is more sensitive than non-parametric testing.

Interpreting the results from the analyses of variance, it becomes evident that the quality of change impact predictions is not impacted by the software tool that is being used for tasks 1 or 3. A similar conclusion can be drawn about the time taken to complete task 3.

The time taken to complete task 1, which adds a part to a requirement, is significantly different for the group that used TRIC. They performed change impact prediction of scenario 1 slower than the other groups.

6.6. Non-parametric testing

Interpreting the results from the χ^2 tests, it becomes evident that the quality of change impact predictions is not impacted by the software tool that is being used for tasks 2, 4 or 5.

The time taken to complete of tasks 4 and 5, who respectively remove a part and remove a whole, is significantly different for the group that used TRIC. For task 4, the TRIC group was slower than the Microsoft Excel group. For task 5, the TRIC group was slower than the IBM Rational RequisitePro group.

The time taken to complete task 2 was indicated to be significantly different for the group that used TRIC by the χ^2 test, but an ensuing post-hoc comparison using a Mann-Whitney U test indicated that this result is a false positive. This false positive may be caused as a result of a low sample size. The χ^2 test compares three groups simultaneously, while the Mann-Whitney U test compares two groups at a time. Indeed, it is known that χ^2 test are conducive to producing false positives with small sample sizes [16].

6.7. Analysis of covariance

The reliability of the covariates was too low to conduct an analysis of variance. This means that the hypothesized covariates do not explain for the difference in results inside of the groups. The question why one participant scores better than the other is thus left unanswered.

Of the strongest covariates, the first three somehow measure the same construct. The completion of a basic requirements engineering course, completion of an advanced requirements engineering course, and months of experience, are in fact all a measure of experience with requirements management. Indeed, statistical testing does detect correlations amongst these variables of medium effect size. Thus, overall experience with requirements management is the largest confirmed covariate.

6.8. Multivariate analysis of variance

The assumption of linearity between the *F*-score of change impact predictions and the time taken to complete them was violated. Hence a multivariate analysis of variance could not be performed.

Previous analyses of variance and non-parametric tests already revealed that there were no differences in *F*-scores between the groups, but that the TRIC group did take longer for four out of five tasks. This supports the results of non-linearity.

One way to explain the longer time taken yet equal *F*-score of the TRIC group is that TRIC is a more intelligent tool. It offers more visualization opportunities and is not as mature as the other software tools. If the benefits of TRIC are to better cope with complexity, then those may only be reaped with an appropriately complex software requirements specification.

6.9. Conclusion

Although the analysis of the change scenarios revealed a diverse feasibility, the fact that the golden standards have a very high degree of reliability proves that the experimental instrumentation is reliable in spite of the lack of theory surrounding change scenarios.

The findings of the statistical analyses of the F-scores and time taken suggest that the TRIC group did not produce better quality change impact predictions, but did take longer to complete their predictions in three out of five cases.

Finally, covariate reliability testing suggests that experience with requirements management is the most influential covariate of all covariates that were tracked.

7. Conclusions and future work

7.1. Summary

The background for this research was to evaluate the impact of TRIC, a software tool that supports the formal requirements relationship types that were developed in the QuadREAD Project, on the quality of change impact predictions. It was hypothesized that using TRIC would positively impact that quality. A quasi-experiment was systematically designed and executed to empirically validate this impact.

A review of existing literature found that the quality of change impact prediction can be measured by the *F*-score and visualized in Precision-Recall graphs and Receiver Operating Characteristics. Less agreement existed on the measurement of change impact prediction effort. This experiment used the *F*-measure and time taken to complete change impact prediction as dependent variables. The visualizations did not convey much information in this experiment due to a small sample size of 21 participants.

The independent variable used in this experiment was the level of tool support. The participants were assigned with either Microsoft Excel, IBM Rational RequisitePro or TRIC to perform change impact prediction for several change scenarios. However, a lack of theory regarding change scenarios caused the influence of change scenarios on the experimental results to be uncontrolled, which is a concern for reliability.

The research design revealed that there were not enough TRIC experts in existence to participate in the experiment. This meant that non-expert participants had to be trained to play the role of expert. This posed two important threats to validity. First, this threatens internal validity because the lecture effect is difficult to control. Second, it threatens external validity because the non-experts may not be representative for experts or even system maintenance engineers in general. This is an inherent problem when attempting to empirically provide a solution validation to new software tools.

The object used in the experiment was the WASP specification, a software requirements specification which was found to be clear and of low complexity. Recognizing the benefit of TRIC to deal with complex specifications yet being unable to acquire one of ample complexity meant that the WASP specification was likely to cause non-significant results. No other public and usable specifications could be found.

A group of experts created a golden standard to compare participants' change impact predictions against. The inter-rater reliability of these golden standards was high, indicating that the experimental instrumentation is reliable in spite of reliability issues concerning the change scenarios.

7.2. Results

The results of this specific experiment do not provide a positive solution validation of TRIC. The following conclusions can be drawn with respect to the combination of participants, change scenarios and software requirements specification that were used in this experiment:

- Null hypothesis 1 stated that the *F*-scores of change impact predictions of system maintenance engineers using TRIC will be equal to or less than those from system maintenance engineers. This null hypothesis was accepted.

- Null hypothesis 2 stated that the time taken to complete change impact predictions of system maintenance engineers using TRIC will be equal to or longer than those from system maintenance engineers not using TRIC. This null hypothesis was also accepted.

No differences in the quality of change impact predictions between using Microsoft Excel, IBM Rational RequisitePro or TRIC were detected. However, using TRIC was detected to lead to slower change impact prediction. The mean statistical power of the tests underlying these conclusions is 54%.

Covariate reliability testing further suggested that experience with requirements management is the most covariate of all covariates, although the way it was constructed in this experiment is not reliable enough to explain any variance in *F*-scores or time taken to complete change impact predictions.

Limitations of this research mean that these results cannot be generalized.

7.3. Limitations

The results of this research are subject to the following limitations:

- **Lack of control over lecture effect.** Participants require training to work with the software tools and play the role of expert. This is difficult to do reliably. First, the setup of the lecture is not fair because the same time is allotted for all three software tools, although RequisitePro and TRIC require more tutoring than Excel. Second, a reliable pre-test and

post-test to measure software tool aptitude and the learning effect of the lecture is not available.

The same problem is known in marketing, where there are no existing consumers of a new product. In Kotler's eight-step process of new product development, it is suggested that concept testing is performed with focus groups. A focus group is defined to be a small sample of typical consumers under the direction of a group leader who elicits their reaction to a stimulus such as an ad or product concept. They are one form of exploratory research that seeks to uncover new or hidden features of markets and can help solve problems. However, focus groups usually suffer from small sample sizes, limited generalizability and Hawthorne effects [35]. The problem-solving and exploratory approaches match that of action research, which seems a more plausible way of validating new software tools, though that is subject to the same challenges as focus group research [72].

- **Low participant representativeness.** There is no strong evidence to assume that master students are representative for actual system maintenance engineers. Although an argument can be made that a sampling of 22 master students in Computer Science and Business Information Technology can be representative for their larger population, the data set contained a sizable number of outliers for which there were no grounds for data set reduction. The experiment should be repeated with different participants to assert external validity.

- **Lack of control over change scenarios.** This research instructs participants to perform change impact prediction on a set of change scenarios. It is likely that change scenarios have influence over the results of change impact predictions, but the lack of theory surrounding change scenarios is a cause of reliability problems. Second, some students raised questions about the rationales in the change scenarios, which may have induced guessing. This limitation is partially offset by the high inter-reliability scores of the golden standards, which indicate that a group of experts interpret the change scenarios reliably and proves the usability of the experimental design if enough experts were available.

- **Small sample size.** The sample size of the research is too small to attain the generally accepted statistical power of 80%. Instead, the statistical power is 56% for the analyses of variance and 52% for the non-parametric tests. If the statistical power increases, then inferences can be made with greater confidence and smaller effects could be detected.

- **Limited comparability of software tools.** No statistical adjustments have been made for the functionality, maturity and usability of either Microsoft Excel, IBM Rational Requi-

sitePro or TRIC. Even though they all feature a traceability matrix, other tools may produce different results. Inferences can only be made with regards to these three tools.

- **Monogamous metrics.** By only using the F-score, it is possible that the quality of change impact predictions is not measured fully and that the measurement of quality is mixed with measurement of the metric. Having more measures of quality would improve the reliability of the results.

- **Low participant reliability.** First, not all participants were as focused on the task as was expected. Second, many were under pressure to complete the experiment. Third, some participants did not check the initially changed requirement as part of their Estimated Impact Set, even though they were instructed to do so both during the lecture and by the web application. This may have led to suboptimal change impact predictions. Using experts instead of master students is not certain to produce more reliable results, because interviews have indicated that the effort of experts also depends on their stake in the project. However, shorter experiments will produce more reliable results [21].

- **Limited research object representativeness.** Specifications other than the WASP specification used can have different complexity in terms of length, structure, ambiguity, completeness and possibly other metrics which were not discussed here. This can influence the impact of using different software tools on the quality of change impact predictions. For example, an intelligent tool such as TRIC is likely to only show its benefits when tasked with a complex software requirements specification. The experiment should be repeated with a diverse set of specifications to evaluate the influence of these attributes.

- **Limited control over environment.** The experiment locations were not comparable in terms of layout or noise. The experiment should be conducted on a location with equal and neutral rooms for the groups.

7.4. Future work

The solution validation of the requirements metamodel and supporting TRIC tool is an open end that is worth pursuing. It is hypothesized that the lack of a positive solution validation by this research can be attributed to the fact that TRIC is a more intelligent software tool and its benefits will only materialize given a sufficiently complex software requirements specification. This is supported by the findings on the earlier experiment on trace approaches, which detected significant results while using a software requirements specification of higher complexity.

To further the solution validation, the following can be recommended:

- Further the state-of-the-art in change scenario theory, so that it is clear how a certain change scenario can impact change impact prediction. Much theory exists on change impact prediction, but not on the elements of change scenarios themselves. The research should be focused on real-world practice, admitting that most real-world changes will not comply to a yet to be determined academic standard. This is required to complete the necessary body of knowledge to setup a controlled experiment.

- Create multiple change scenarios of the same class. This research used an improvised classification according to the type of requirements change in terms of its part or whole. The effect of this classification could not be tested because only one class of change scenarios was represented twice.

- Find a number of real-world software requirements specifications of high complexity. As with change scenario theory, there is no generally accepted criterion for what constitutes complexity, although raw indices such as page count, requirements count and tree impurity will provide a strong argument. If these specifications cannot be collected from the Quad-READ Project partners, then it is worthwhile asking governmental institutions to participate in academic research, possibly under non-disclosure agreement.

- Consider organizing an online experiment, where experts can participate from behind their own computer. This allows more time for experimentation, because the experiment can be split up into several time slots which can stretch multiple days. It also lowers the barrier to entry to participate. Given a large enough sample size, the lack of environmental control will be corrected for by randomization.

- Consider organizing multiple action research projects, where researchers can apply the techniques in practical cases that are currently running with clients. As a precondition, it should be accepted that action research is cyclical and that TRIC must evolve as part of the cases. Give a large enough amount of action research iterations, a strong argument for generalizability may be found.

Not related to this solution validation, but a recommendation for future work nonetheless is to research the impact of classes of software tools with the same intelligence on the quality of change impact predictions. It can answer the question if it makes sense to have more intelligent software tooling during software maintenance. This requires the creation of a classification scheme for levels of software tool intelligence, which currently does not exist.

8. Glossary

Action research: An interventionist research method that is dedicated to the development of knowledge useful to both research and practice [41].

Actual Impact Set: Set of objects that were actually modified as the result of performing the change [6].

Case study: An empirical inquiry that (1) investigates a contemporary phenomenon in depth and within its real-life context, especially when the boundaries between phenomenon and context are not clearly evident; (2) copes with the technically distinctive situation in which there will be many more variables of interest than data points [72].

Cause: A variable that produces an effect or result [52].

Change impact analysis: The activity of identifying what to modify to accomplish a change, or of identifying the potential consequences of a change [6].

Consistency: In software requirements specifications, refers to internal consistency as measured by the number of conflicts between subsets of individual requirements [59].

Consistency checking: The activity to identify the relationship whose existence causes a contradiction [25].

Construct validity: The degree to which inferences are warranted from the observed persons, settings and cause-and-effect operations sampled within a study to the constructs that these samples represent [52].

Discovered Impact Set: Set of objects that were not estimated by the change impact analysis to be affected, but were affected during performing the change [6].

Empirical methods: The use of empirical methods to ascertain facts, that is, methods that are based on experience or observation of the world [17].

Estimated Impact Set: Set of objects that are estimated to be affected by the change [6].

Experiment: To explore the effects of manipulating a variable [52].

External validity: The validity of inferences about whether the causal relationship holds over variations in persons, settings, treatment variables and measurement variables [52].

False Positive Impact Set: Set of objects that were estimated by the change impact analysis to be affected, but were not affected during performing the change [6].

Inferencing: In requirements engineering, the activity of deducing new relationships based solely on the relationships which a requirements engineer has already specified [25].

Inter-rater reliability: The level of agreement between expert ratings [30].

Internal validity: The validity of inferences about whether the relationship between two variables is causal [52].

Metamodel: A model of modeling language [36].

Model: Represents a part of the reality called the object system and is expressed in a modeling language. A model provides knowledge for a certain purpose that can be interpreted in terms of the object system [36].

QuadREAD Project: Quality-Driven Requirements Engineering and Architectural Design. A joint research project of the Software Engineering Group and Information Systems Group at the University of Twente [50].

Quality: (1) The degree to which a system, component, or process meets specified requirements; (2) The degree to which a system, component, or process meets customer or user needs or expectations [58].

Quasi-experiment: An experiment in which units are not randomly assigned to conditions [52].

Random assignment: In an experiment, any procedure for assigning units to conditions based on chance, with every unit having a nonzero probability of being assigned to each condition [52].

Requirement: (1) A condition or capability needed by a user to solve a problem or achieve an objective; (2) A condition or capability by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents; (3) A documented representation of a condition or capability as in (1) or (2) [58].

Requirements management: The process of understanding and controlling changes to requirements [55].

Requirements phase: The period of time in the software life cycle during which the requirements for a software product are defined and documented [58].

Requirements review: A process or meeting during which the requirements for a system, hardware item, or software item are presented to project personnel, managers, users, customers or other interested parties for comment or approval [58].

Requirements specification: A document that specifies the requirements for a system or component [58].

Requirements validation: Checking that requirements meet the real demands of stakeholders [9].

Research design: Guides the investigator in the process of collecting, analyzing and interpreting observations. It is a logical model of proof that allows the researcher to draw inferences concerning causal relations among the variables under investigation [45].

Scenario: A small story with a vivid illustration of the work area or a specific case of a task [39].

Software requirements specification: Documentation of the essential requirements (functions, performance, design constraints and attributes) of the software and its external interfaces [58].

Software tool: A computer program used in the development, testing, analysis, or maintenance of a program or its documentation [58].

Specification: A document that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system or component and often the procedures for determining whether these provisions have been satisfied [58].

Statistical conclusion validity: The validity of inferences about the correlation between treatment and outcome: whether the presumed cause and effect covary and how strongly they covary [52].

Trace: To establish a relationship between two or more products of the development process [58].

Traceability: The degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another [58].

Validity: The truth of, or correctness of, or degree of support for an inference [52].

9. References

1. Abbattista, F., et al. *An Experiment on the Effect of Design Recording on Impact Analysis*. in *International conference on Software Maintenance*. 1994. Victoria, BC: IEEE Computer Society Press. pp. 253-259.

2. Abma, B.J.M., *Evaluation of requirements management tools with support for traceability-based change impact analysis*. 2009, Master's thesis, University of Twente: Enschede.

3. Ajila, S., *Software maintenance: An approach to impact analysis of object change.* Software - Practice and Experience, 1995. **25**(10): pp. 1155-1181.

4. Alexander, I. & Robertson, S., *Understanding project sociology by modeling stakeholders.* IEEE Software, 2004. **21**(1): pp. 23-27.

5. Argyris, C. & Schön, D., *Participatory Action Research and Action Science Compared*, in *Participatory Action Research*, W.F. Whyte, Editor. 1991, Sage: Newbury Park, New Jersey. pp. 85-96.

6. Arnold, R.S. & Bohner, S.A. *Impact Analysis - Towards A Framework for Comparison*. in *Conference on Software Maintenance*. 1993. Montreal, Quebec: IEEE Computer Society. pp. 292-301.

7. Ash, S., *MoSCoW Priorisation Briefing Paper*. 2007, DSDM Consortium.

8. Babar, M.A. & Gorton, I. *Comparison of Scenario-Based Software Architecture Evaluation Methods*. in *11th Asia-Pacific Software Engineering Conference*. 2004. Busan, Korea: IEEE Computer Society. pp. 600-607.

9. Baskerville, R. & Wood-Harper, A.T., *Diversity in information systems action research methods.* European Journal of Information Systems, 1998. **7**: pp. 90-107.

10. Baskerville, R.L. & Wood-Harper, A.T., *A critical perspective on action research as a method for ifnormation systems research.* Journal of Information Technology, 1996. **11**(3): pp. 235-246.

11. Bengtsson, P. & Bosch, J. *Architecture Level Prediction of Software Maintenance*. in *Third European Conference on Software Maintenance and Reengineering*. 1999. Amsterdam, The Netherlands: IEEE Computer Society. pp. 139-147.

12. Bohner, S.A. & Arnold, R.S., *Software Change Impact Analysis*. 1996, Los Alamitos, CA: IEEE Computer Society Press.

13. Cohen, J., *Statistical power analysis for the behavioral sciences*. 1988, Hillsdale, New Jersey: Erlbaum.

14. Dahlstedt, Å.G. & Persson, A., *Requirements Interdependencies: State of the Art and Future Challenges*, in *Engineering and Managing Software Requirements*, A. Aurum and C. Wohlin, Editors. 2005, Springer: Berlin Heidelberg. pp. 95-116.

15. Davis, J. & Goadrich, M. *The Relationship Between Precision-Recall and ROC Curves*. in *23rd International Conference on Machine Learning*. 2006. Pittsburgh, Pennsylvania: ACM. pp. 233-240.

16. Dawson, B. & Trapp, R.G., *Basic & Clinical Biostatistics*. 4th ed. 2004: McGraw-Hill.

17. Dooley, D., *Social research methods*. 4th ed. 2001, Upper Saddle River, New Jersey: Prentice-Hall.

18. Ebben, P., et al., *Requirements for the WASP Application Platform*, in *WASP/D2.1*. 2002, Telematica Instituut: Enschede, The Netherlands.

19. Everett, D.F., et al. *Conversion From Tree to Graph Representation of Requirements*. 2009 [cited 2009 May 10]; Available from: http://www.techbriefs.com/component/content/article/5059.

20. Fawcett, T., *ROC Graphs: Notes and Practical Considerations for Researchers*. 2004, Technical report, HP Laboratories: Palo Alto, California.

21. Fitz-Gibbon, C.T. & Vincent, L., *Difficulties regarding Subject Difficulties: Developing Reasonable Explanations for Observable Data.* Oxford Review of Education, 1997. **23**(3): pp. 291-298.

22. Fleiss, J.L., Levin, B. & Paik, M.C., *Statistical Methods for Rates and Proportions*. 3rd ed. 2003: Wiley-Interscience.

23. Garson, D. *Quantitative Research in Public Administration*. [cited 2009 October 6]; Available from: http://faculty.chass.ncsu.edu/garson/PA765/index.htm.

24. Goknil, A., *Tutorial: Requirements Relations and Definitions with Examples*. 2008, Internal document, University of Twente: Enschede.

25. Goknil, A., et al., *Semantics of Trace Relations in Requirements Models for Consistency Checking and Inferencing*. 2009, to be published in Software and Systems Modeling, University of Twente: Enschede.

26. Gotel, O. & Finkelstein, A., *An Analysis of the Requirements Traceability Problem*, in *IEEE International Conference on Requirements Engineering*. 1994, IEEE Computer Society Press: Los Alamitos, California. pp. 94-101.

27. IBM, *Rational RequisitePro*.

28. International Organization for Standardization, *ISO 13407*. 1999.

29. International Organization for Standardization, *ISO/IEC 12207*. 2008.

30. Jedlitschka, A. & Pfahl, D. *Reporting Guidelines for Controlled Experiments in Software Engineering*. in *International Symposium on Empirical Software Engineering*. 2005. Noosa Heads, Australia: IEEE Computer Society Press. pp. 95-104.

31. Kääriäinen, J., et al., *Improving Requirements Management in Extreme Programming with Tool Support – an Improvement Attempt that Failed*, in *30th EUROMICRO Conference*. 2004, IEEE Computer Society Press: Rennes, France. pp. 342-351.

32. Kampenes, V.B., et al., *A systematic review of quasi-experiments in software engineering.* Information and Software Technology, 2009. **51**: pp. 71-82.

33. Katz, S., et al., *Gender and Race in Predicting Achievement in Computer Science*, in *IEEE Technology and Society Magazine*. 2003.

34. Koolwaaij, J.W., *Interview about the WASP specification*, R.S.A. van Domburg and A. Goknil, Editors. 2009: Enschede.

35. Kotler, P., et al., *Principles of Marketing*. 5th European ed. 2008, Essex: Pearson Education Limited.

36. Kurtev, I., *Adaptability of Model Transformations*. 2005, PhD thesis, University of Twente: Enschede.

37. Kurtev, I., *Requirements for the Course Management System*. 2009, Internal document, University of Twente: Enschede.

38. Lassing, N., et al., *Experiences with ALMA: Architecture-Level Modifiability Analysis.* The Journal of Systems and Software, 2002(61): pp. 47-57.

39. Lauesen, S., *Software Requirements: Styles and Techniques*. 2002, Harlow: Pearson Education Limited.

40. Leech, N.L., Barrett, K.C. & Morgan, G.A., *SPSS for Intermediate Statistics: Use and Interpretation*. 2004, New York: Lawrence Erlbaum Associates.

41. Lindgren, R., Henfridsson, O. & Schultze, U., *Design Principles for Competence Management Systems: A Synthesis of an Action Research Study.* MIS Quarterly, 2004. **28**(3): pp. 435-472.

42. Luger, G.F., *Artificial Intelligence - Structures and Strategies for Complex Problem Solving*. 2005, Essex, England: Pearson Education Limited.

43. Mäder, P., Gotel, O. & Philippow, I. *Enabling Automated Traceability Maintenance through the Upkeep of Traceability Relations*. in *5th European Conference on Model Driven Architecture*. 2009. Enschede, The Netherlands: Springer. pp. 174-189.

44. Mockus, A., et al., *On Measurement and Analysis of Software Changes*. 1999, Bell Laboratories.

45. Nachmias, D. & Nachmias, C., *Research methods in the social sciences*. 1992, New York: St. Martin's.

46. Nonaka, I., *A Dynamic Theory of Organizational Knowledge Creation.* Organization Science, 1994. **5**(1): pp. 14-37.

47. Object Management Group, *SysML Version 1.1*. 2008.

48. Osborne, J., *Notes on the use of data transformations.* Practical Assessment, Research & Evaluation, 2002. **8**(6).

49. Pallant, J., *SPSS Survival Manual: A Step By Step Guide to Data Analysis Using SPSS for Windows*. 2001, Buckingham, UK: Open University Press.

50. QuadREAD Project. *QuadREAD Project - Project Description*. 2009 [cited 2009 March 14]; Available from: http://quadread.ewi.utwente.nl/index.php?option=com_content&task=view&id=13&Itemid=29.

51. Robertson, J. & Robertson, S., *Volere Requirements Specification Template*. 2007, The Atlantic Systems Guild.

52. Shadish, W.R., Cook, T.D. & Campbell, D.T., *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*. 2002, Boston, New York: Houghton Mifflin Company.

53. Shrout, P.E. & Fleiss, J.L., *Intraclass Correlations: Uses in Assessing Rater Reliability.* Psychological Bulletin, 1979(2): pp. 420-428.

54. Sjøberg, D.I.K., et al. *Conducting Realistic Experiments in Software Engineering*. in *International Symposium on Empirical Software Engineering*. 2002. Nara, Japan: IEEE Computer Society Press. pp. 17-26.

55. Sommerville, I., *Software Engineering*. 7th ed. International Computer Science Series. 2004, Essex, England: Pearson Education.

56. Stevens, J., *Applied multivariate statistics for the social sciences*. 3rd ed. 1996, Mayway, New Jersey: Lawrence Erlbaum.

57. Taylor, M.J., Wade, S. & England, D., *Informing IT system Web site design through normalisation.* Internet Research: Electronic Networking Applications and Policy, 2003. **13**(5): pp. 342-355.

58. The Institute of Electrical and Electronics Engineers, *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries*. 1991, New York: Institute of Electrical and Electronics Engineers.

59. The Institute of Electrical and Electronics Engineers, *IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications*. 1998: New York.

60. Uebersax, J. *Statistical Methods for Rater and Diagnostic Agreement*. 2009 [cited 2009 October 6]; Available from: http://www.john-uebersax.com/stat/agree.htm.

61. van den Berg, K.G., *Change Impact Analysis of Crosscutting in Software Architectural Design*, in *Architecture-Centric Evolution*. 2006, University of Groningen: Nantes, France. pp. 1-15.

62. van Lamsweerde, A. *Goal-Oriented Requirements Engineering: A Roundtrip from Research to Practice*. in *12th IEEE International Requirements Engineering Conference*. 2004. Kyoto, Japan: IEEE Computer Society. pp. 4-7.

63. van Lamsweerde, A., Darimont, R. & Letier, E., *Managing Conflicts in Goal-Driven Requirements Engineering.* IEEE Transactions on Software Engineering, 1998. **24**(11): pp. 908-926.

64. Veldhuis, J.-W., *Tool support for a metamodeling approach for reasoning about requirements*. 2009, Master's thesis, University of Twente: Enschede.

65. Vicente-Chicote, C., Moros, B. & Toval, A., *REMM-Studio: an Integrated Model-Driven Environment for Requirements Specification, Validation and Formatting.* Journal of Object Technology, 2007. **6**(9): pp. 437-454.

66. von Knethen, A. *A Trace Model for System Requirements Changes on Embedded Systems*. in *International Conference on Software Engineering*. 2001. Vienna, Austria: ACM. pp. 17-26.

67. Weilkiens, T., *Systems engineering with SysML/UML: modeling, analysis, design*. 2006, Burlington, MA: Morgan Kaufmann Publishers.

68. Wieringa, R. *Design Science as Nested Problem Solving*. in *4th International Conference on Design Science Research in Information Systems and Technology*. 2009. Philadelphia, Pennsylvania: ACM.

69. Wieringa, R. & Heerkens, J., *The methodological soundness of requirements engineering papers: A conceptual framework and two case studies.* Requirements Engineering Journal, 2006. **11**(4): pp. 295-307.

70. Wieringa, R. & Heerkens, J., *Designing Requirements Engineering Research*, in *Fifth International Workshops on Comparative Evaluation in Requirements Engineering*. 2007, IEEE.

71. Wolf, R.M. *The Validity and Reliability of Outcome Measure*. in *Monitoring the Standards of Education*. 1994. Oxford: Pergamon.

72. Yin, R.K., *Case Study Research: Design and Methods*. 4th ed. Applied Social Research Methods, ed. L. Bickman and D.J. Rog. 2009, Thousand Oaks, California: SAGE Publications.

73. Zelkowitz, M.V. & Wallace, D.R., *Experimental Models for Validating Technology.* Computer, 1998. **31**(5): pp. 23-31.

A. Interviews

1. Introduction

Two interviews were conducted; one with an academic from the Information Systems group at the University of Twente and one with three industry experts working with Capgemini. This appendix reports on these interviews.

2. Goal

The goal of the interviews was two-fold. First, to attain golden standards for the experiment from these experts. Second, to elicit how experts deal with change impact prediction by performing think-aloud exercises. The researcher provides a change impact prediction task to the interviewee, asking him to perform the task while thinking aloud and observing his actions and lines of reasoning.

3. Preparation

The interviewees were sent the WASP requirements specification in advance and informed that they would be tasked with predicting a change scenario. They were told that the interview would last approximately half a day.

The interview was structured around the following questions:

1. What is your occupation?
2. What is your work experience?
3. How are you involved in requirements engineering?
4. How are you involved in change management?
5. How did you prepare for this interview?
6. How did you study the WASP specification?
7. Which items in the document are unclear to you?
8. Explaining the task to perform: which points in the procedure are unclear to you?
9. Providing the change scenarios from the experiment one-by-one:
    1. Which elements in the change scenario are unclear to you?
    2. Which requirements will be impacted by this change?

3. How did you come to this conclusion, step-by-step?

4. Execution

The interviews did not strictly follow the structure mentioned above. The interviewees did not have enough time to perform the change impact prediction, or were not willing to read through the entire WASP specification, in spite of the advance announcement. Because that meant the interview goal could not be attained, the interview was changed on the fly to let the interviewees tell as much as they could about their experiences with change impact prediction.

5. Information systems academic

Background

An interview was held with an assistant professor with the Information Systems group at the University of Twente. Both this research group and this specific assistant professor are involved in the QuadREAD Project. He has worked at the university since 1998.

The assistant professor indicates that he spends 75% of his time lecturing and 50% of his time researching, for a total of 125%. He lectures courses on information systems, systems for cooperative work and business process modeling. He used to lecture a course on requirements engineering and is a practical supervisor for a software engineering course. Finally, he supervisors many master's projects.

The assistant professor supervises PhD students from the Information Systems group in the QuadREAD Project. He is also involved in a research project called U-Care, which aims to develop a service platform for personalized services. He indicates that there is an overlap with the WASP project, which is also concerned with the context and behavior of an actor. Within the U-Care project, he is tasked with eliciting the requirements in terms of stakeholders, scenarios, pilot studies and prototypes.

The assistant professor indicates to have little experience with change management or industrial experience with requirements engineering.

Reading the WASP specification

The assistant professor indicates that he finds the WASP specification to be clear. He finds it well structured. He began reading the scenarios, because he finds individual requirements to provide too little context.

He read two scenarios globally to get an overview. He commends the text-baed style of the change scenarios, indicating that this is what is also taught and used in research at the university.

He browsed over the requirements but did not check for completeness or evolvability. Instead, he predominantly looked into their style and structure. He notes that the WASP specification has attempted to use the MoSCoW approach for classifying requirements, but that this has not been applied rigorously.

He further noted that the non-functional requirements were not traced to the use cases. He finds them to be so vague that they could be removed just as well. He considers the level of detail of the other requirements insufficient for implementation.

Think-aloud exercise

The assistant professor is asked to perform change impact prediction for task 4 (REQ_NAV_003). He proceeds as follows:

1. He browses through the printed WASP specification to find REQ_NAV_003. He realizes that he can use the PDF version to search for it.
2. He searches the PDF for REQ_NAV_003. He finds it on page 44, noticing that there are spelling mistakes in the document.
3. He interprets the scenario. He uses the rationale to provide meaning to the change. He repeats his understanding of the scenario.
4. He inspects REQ_NAV_003 to see if is traced to other requirements. He first looks for sub- or super-requirements, but cannot find mention of them. He does notice that it is traced to use case UC_NAV_001.
5. He inspects requirements that are near REQ_NAV_003 in the document.
6. He stops his inspection of neighboring requirements and starts searching in the use cases. He believes that the functionality deals with location selection.
7. Once at the use cases, he is unable to find the navigation use cases. He thinks that they might be with Personalized Dynamic Navigation, but it turns out to be not so.
8. He finds the navigation functionality specified as “POI”. He browses to the use case text and it seems relevant. He discovers that there is a requirement REQ_TOR_001 for a user to find points of interest.

9. He looks up REQ_TOR in the PDF and interprets the requirement. He finds it ambiguous: are the items in the description related as AND or as OR? Do they have global coverage?
10. He now considers if the requested change is an addition to this requirement. He marks the requirement as impacted and considers REQ_TOR_002 to be a sub-requirement of REQ_TOR_001, therefore also being impacted.

Reflection on exercise

The assistant professor indicates that he has performed the think-aloud exercise mostly on intuition. To be absolutely sure of his prediction, he would manually inspect all requirements. That level of certainty is not required now, so he does not do that.

As a shortcut he would inspect all of the headings in the requirements chapter to quickly find potentially impacted groups of requirements. He notes that there is no tracing from use cases downward to requirements, so it is not very useful to scan the use cases as a shortcut.

He adds that such exhaustive searches are not scalable. Therefore he would suggest having full top to bottom tracing and a good intuition and knowledge about the system, though predictions will not be failsafe.

He does consider having a high degree of traceability to be the best support in predicting change impact, yet also believes that consulting a document expert is the best overall method. Traces do not provide an ultimate truth, so document insight will always be required.

He suggest adding traces between groups of requirements to quickly navigate between groups of related functionality when doing change impact prediction.

6. Industry experts at Capgemini

Background

Mr. A is a systems analysis and RUP process analyst. He has also provided training to the development street on the topics of general processes, requirements engineering and configuration and change management. Before that he was involved in SAP implementations and as a Microsoft Dynamics sales and project manager.

Mr. B is concerned with controlled migrations in Capgemini. Before that he worked with the automation department of Stork as programmer. He later worked for the editorial system of a large newspaper. Going back to Stork, he has been a database administrator, project leader and

manager of application support. In 1995 his work was outsourced to the RCC where he participated in Y2K projects, offshoring to India and Euro conversion. He was involved in doing controlled migrations and testing. He has always been part technician and part user. He is an active participant in research & development projects with universities.

Mr. C has been working at Capgemini for nine years. Before that he has fulfilled positions as service manager and operations manager in a local government. At Capgemini, he started to develop hardware for an ordering portal. In 2006 he moved to application management. He is now in charge of expanding the reconstruction service. He has always worked with software from a technical perspective. His experience with change management is predominantly within the field of operations management.

Reading the WASP specification

Mr. A regards the document as refreshing and modern. The finds the use cases to be agile, well-connected and extensive. The document provides him with a clear view. He does wonder about the system goal, industry and intended reader. He notes that there are rather diverse focuses on functionality. He partly read the introduction, scenarios and use cases.

At this stage, Mr. A is unable to say or do much with the document. He indicates to have performed a rather informal reviewing process when a formal process is required for proper change impact prediction. He is unsure about the meaning of preconditions and assumptions and the level of tracing. He says to be somewhat conservative because the document is so skinny.

Mr. A wonders about the design decisions and architectural constraints. A full tracing of requirements, architectural design and code is necessary to make informed decisions.

Mr. B does not understand what WASP stands for, although he regards the specification with enthusiasm. He read it up to and including the use cases. He finds it a decent document and certainly in comparison to that of some legacy systems he has worked with. He finds the specification to be complete and usable for function point analysis. He read the introduction, scenarios and use cases and browsed through the requirements.

Mr. B is mostly familiar with legacy documents, which he is used to scan for keywords. He then relates these keywords himself within a visual tool. He would then reduce the requirements to function points. He has not done so for this document.

Mr. C considers the requirements to be rather high-level. Differences between the platform and application are unclear. Finally, he finds that the requirements have a diverse level of detail. He mostly read through the requirements chapter.

Costs and benefits

Mr. A notes that this traceability often costs to maintain than to cope with imperfections in change impact predictions. An important question, he adds, is whether or not much change is to be expected. Often, requirements only turn out to be impacted in the long run and at that point there may be ample budget to cope with that change.

All three experts add that the type and process methodology of the project are very important. New-built projects are treated different from acquired existing projects. At Capgemini, there are different business units for new development and maintenance. The focus for new development is becoming a "shipping focus" more and more. Unless there is a process methodology that promotes traceability and sensitivity analysis and an idea of the maintenance team, documentation may lag behind. Overall, the availability of experts is more important than the availability of traces when transferring new developments to maintenance.

Mr. A mentions that a reason for adding traceability separate from making better quality predictions is to mitigate transferability risks. Proper product and process documentation that is transferrable reduces the risk of a person, who may be injured or leave the company.

Still, all three experts agree that experts are the most useful sources of knowledge for making change impact predictions and performing system evolution. They are very intimate with the code and are more cost effective than having full traceability. On the other hand, expert judgements are not verifiable by management and here traceability plays an important communication and justification function. Traces can be used to explain management the impact of a certain change.

Mr. C adds that changes over time make a system stiff and less changeable. Mr. A agrees and believes the cause to be shortcuts taken due to budgetary constraints. Without a proper standard and methodology for meeting software quality, developers and architects become less critical of the quality. Critical changes should be reviewed together with experts. Here, traces are useful in the reviewing phase of an initial change estimate. Experience has shown that initial estimates are always off, and a second opinion can improve quality of software and predictions.

7. Conclusions

The interviews did not yield any golden standards, but did provide several insights:

- The academic expert regarded use cases to be very important to relate groups of requirements to each other. He considers traceability between groups of requirements to be a useful addition to discover related requirements.
- All experts agree that the availability of experts in change impact prediction is often more important then the availability of traces. Traces can play an important communication function to justify, verify and improve the expert prediction.
- All experts agree that if traces are added, then there should be full traceability from top to bottom. The benefits of having this level of traceability may not outweigh the costs, unless there is some process or quality commitment to having a high degree of documentation.
- The experts from industry agree that the capturing of design decisions and traces to architecture design and code are required for making sound change impact predictions based on traces.

B. Tasks

B.1. Introduction

This appendix lists the tasks that were provided to the participants.

B.2. Warming up (REQ_BDS_007)

Modify REQ_BDS_007 "When changes are discovered in the status and/or location of a user's buddy, the WASP platform MUST sent out notifications according to the alerts set by the user (see also REQ_NOT_006)."

To: "When changes are discovered in the status of a user's buddy, the WASP platform MUST sent out notifications according to the alerts set by the user (see also REQ_NOT_006)."

Rationale: All functionality to track the location of users is removed from the WASP platform.

B.3. Task 1 (REQ_PHN_001)

Modify REQ_PHN_001 "A WASP application SHALL be able to setup a phone call connection using the 3G Platform via the WASP platform."

To: "The WASP application SHALL be able to setup phone call connections and video chat connections using the 3G Platform via the WASP platform."

Rationale: Video chat functionality is added to the WASP application.

B.4. Task 2 (REQ_SPM_004)

Modify REQ_SPM_004 "The platform must be able to store POI and service profiles."

To: "The platform must be able to store POI profiles."

Rationale: All service profile functionality is removed from the platform.

B.5. Task 3 (REQ_MAP_002)

Modify REQ_MAP_002 "The WASP platform MUST be able to show the location of users on a map."

To: “The WASP platform MUST be able to show the location of users on a map by marking them with a circle.”

Rationale: Circles are a common way of depicting locality on a map, and so the WASP platform should also use that for usability reasons.

B.6. Task 4 (REQ_NAV_003)

Modify REQ_NAV_003 “The WASP platform SHOULD be possible to determine the location of touristic attractions close to a calculated route.”

To: “The WASP platform SHOULD be possible to determine the location of touristic attractions.”

Rationale: Determining locations of touristic attractions should not be limited to closeness to a calculated route. Users may be willing to deviate from a calculated route to visit an attraction.

B.7. Task 5 (REQ_TOR_001)

Delete requirement REQ_TOR_001 “The WASP platform SHALL provide functionality to find points of interest that match the user’s explicit need and obey the restrictions following from the user’s profile and current context.”

Rationale: All point of interest functionality is removed from the WASP platform.

C. Group matching

C.1. Introduction

This appendix reports how the participants were divided over three groups, first at random, second by matching the groups as closely as possible and the final turnout.

C.2. Coding

Abbreviation	Description	Coding	
ID	Student ID	n/a	
G	Gender	0	Female
		1	Male
LED	Level of completed education	0	Bachelor or lower
		1	Bachelor of Science or higher
CED	Country of completed education	0	The Netherlands
		1	Other country
MSC	Current educational program	0	Computer Science
		1	Business & IT
ARE	Completed advanced requirements engineering course	0	No
		1	Yes
RME	Requirements management experience	0	fewer than 3 months
		1	3 months or more

C.3. Pre-experiment randomized

Excel

ID	G	LED	CED	MSC	BRE	ARE	RME
21628	I	I	O	O	O	O	O
37230	I	I	O	I	I	O	O
71587	I	I	O	I	I	O	I
89990	I	I	O	O	O	O	O
196614	I	O	O	I	O	O	O
206571	I	I	I	I	I	O	O
211494	I	I	I	I	O	O	O
Total	7	6	2	5	3	0	1

IBM Rational RequisitePro

ID	G	LED	CED	MSC	BRE	ARE	RME
206547	O	O	I	I	O	O	O
206954	I	I	I	I	I	O	I
207802	O	I	O	I	O	O	O
208558	I	O	O	I	O	O	I
211656	I	I	I	O	O	O	I
214787	I	I	I	O	O	O	O
Total	4	4	4	4	1	0	3

TRIC

ID	G	LED	CED	MSC	BRE	ARE	RME
26948	I	I	O	I	I	O	I
39144	I	O	O	I	O	O	I
134872	I	O	O	O	O	O	I
204048	I	I	I	O	O	O	O
205451	I	O	O	I	I	I	I
205494	I	O	O	O	O	O	O
205605	I	I	I	I	I	O	I
Total	7	3	2	4	3	1	5

C.4. Pre-experiment tuned

Excel

ID	G	LED	CED	MSC	BRE	ARE	RME
37230	I	I	O	I	I	O	O
196614	I	O	O	I	O	O	O
134872	I	O	O	O	O	O	I
204048	I	I	I	O	O	O	O
205605	I	I	I	I	I	O	I
206954	I	I	I	I	I	O	I
207802	O	I	O	I	O	O	O
211605	I	I	I	I	O	O	O
Total	7	6	4	6	3	0	3

IBM Rational RequisitePro

ID	G	LED	CED	MSC	BRE	ARE	RME
71587	I	I	O	I	I	O	I
211494	I	I	I	I	O	O	O
26948	I	I	O	I	I	O	I
39144	I	O	O	I	O	O	I
205494	I	O	O	O	O	O	O
214787	I	I	I	O	O	O	O
Total	6	4	2	4	2	0	3

TRIC

ID	G	LED	CED	MSC	BRE	ARE	RME
89990	I	I	O	O	O	O	O
206571	I	I	I	I	I	O	O
205451	I	O	O	I	I	I	I
208558	I	O	O	I	O	O	I
211656	I	I	I	O	O	O	I
21628	I	I	O	O	O	O	O
206547	O	O	I	I	O	O	O
Total	6	4	3	4	2	1	3

C.5. Experiment final

Excel

ID	G	LED	CED	MSC	BRE	ARE	RME
37230	I	I	O	I	I	O	O
196614	I	O	O	I	O	O	O
134872	I	O	O	O	O	O	I
204048	I	I	I	O	O	O	O
205605	I	I	I	I	I	O	I
206954	I	I	I	I	I	O	I
207802	O	I	O	I	O	O	O
211605	I	I	I	I	O	O	O
Total	7	6	4	6	3	0	3

IBM Rational RequisitePro

ID	G	LED	CED	MSC	BRE	ARE	RME
71587	I	I	O	I	I	O	I
211494	I	I	I	I	O	O	O
26948	I	I	O	I	I	O	I
39144	I	O	O	I	O	O	I
205494	I	O	O	O	O	O	O
Total	5	3	1	4	2	0	3

TRIC

ID	G	LED	CED	MSC	BRE	ARE	RME
89990	I	I	O	O	O	O	O
206571	I	I	I	I	I	O	O
205451	I	O	O	I	I	I	I
208558	I	O	O	I	O	O	I
211656	I	I	I	O	O	O	I
21628	I	I	O	O	O	O	O
206547	O	O	I	I	O	O	O
Y [a]	I	I	I	I	O	O	O
Total	7	5	4	5	2	1	3

a. Participant without a student ID

D. Golden standards

D.1. Introduction

This appendix lists the golden standards that were created by the expert from Novay and researchers from the University of Twente. One golden standard was created for each task from appendix C. "1" indicates that the requirement was deemed to be impacted as part of the change; "0" indicates the contrary.

D.2. Task 1 (REQ_PHN_001)

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_SCH_001	0	0	1	0	0
REQ_WBS_001	0	0	0	0	0
REQ_WBS_002	0	0	0	0	0
REQ_WBS_003	0	0	0	0	0
REQ_WBS_004	0	1	0	0	0
REQ_WBS_005	0	0	0	0	0
REQ_MUS_001	0	0	0	0	0
REQ_MUS_002	0	0	0	0	0
REQ_MUS_003	0	0	0	0	0
REQ_MUS_004	0	0	0	0	0
REQ_MUS_005	0	0	0	0	0
REQ_MUS_006	0	0	0	0	0
REQ_MUS_007	0	0	0	0	0
REQ_PAY_001	0	0	0	0	0
REQ_PAY_002	0	0	0	0	0
REQ_TOR_001	0	0	0	0	0
REQ_TOR_002	0	0	0	0	0
REQ_TOR_003	0	0	0	0	0
REQ_BDS_001	0	0	0	0	0
REQ_BDS_002	0	0	0	0	0
REQ_BDS_003	0	1	0	0	0
REQ_BDS_004	1	1	0	1*	1
REQ_BDS_005	0	0	0	0	0
REQ_BDS_006	0	0	0	0	0

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_BDS_007	o	o	o	o	o
REQ_PHN_001	I	I	I	I	I
REQ_USR_001	o	o	o	o	o
REQ_USR_002	o	o	o	o	o
REQ_USR_003	o	o	o	o	o
REQ_USR_004	o	o	o	o	o
REQ_USR_005	o	o	o	o	o
REQ_USR_006	o	o	o	o	o
REQ_USR_007	o	o	o	o	o
REQ_RES_001	o	o	o	o	o
REQ_RES_002	o	o	o	o	o
REQ_RES_003	o	o	o	o	o
REQ_RES_004	o	o	o	o	o
REQ_RES_008	o	o	o	o	o
REQ_RES_009	o	o	o	o	o
REQ_MAP_001	o	o	o	o	o
REQ_MAP_002	o	o	o	o	o
REQ_MAP_004	o	o	o	o	o
REQ_MAP_005	o	o	o	o	o
REQ_MAP_006	o	o	o	o	o
REQ_MAP_007	o	o	o	o	o
REQ_MAP_008	o	o	o	o	o
REQ_MAP_009	o	o	o	o	o
REQ_NAV_001	o	o	o	o	o
REQ_NAV_002	o	o	o	o	o
REQ_NAV_003	o	o	o	o	o
REQ_NAV_004	o	o	o	o	o
REQ_NOT_001	o	o	o	o	o
REQ_NOT_002	o	I	I	I*	I
REQ_NOT_003	o	o	o	o	o
REQ_NOT_004	o	o	o	o	o
REQ_NOT_006	o	o	o	o	o
REQ_NOT_007	o	o*	I	o	o
REQ_NOT_009	o	o	o	o	o
REQ_NOT_010	o	o	o	o	o

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_LGN_001	0	0	0	0	0
REQ_LGN_002	0	0	0	0	0
REQ_LGN_003	0	0	0	0	0
REQ_SPM_001	0	0	0	0	0
REQ_SPM_002	0	0	0	0	0
REQ_SPM_003	0	0	0	0	0
REQ_SPM_004	0	0	0	0	0
REQ_SPM_005	0	0	0	0	0
REQ_SPM_006	0	0	0	0	0
REQ_SPM_007	0	0	0	0	0
REQ_NF_001	0	0	0	0	0
REQ_NF_002	0	0	0	0	0
Impacted set size	2	5	4	3	3

D.3. Task 2 (REQ_SPM_004)

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_SCH_001	0	0	0	0	0
REQ_WBS_001	0	1	1	1	1
REQ_WBS_002	0	0	0	0	0
REQ_WBS_003	0	1	0	0	0
REQ_WBS_004	1	1	1	1	1
REQ_WBS_005	0	0	0	0	0
REQ_MUS_001	0	0	0	0	0
REQ_MUS_002	0	0	0	0	0
REQ_MUS_003	0	0	0	0	0
REQ_MUS_004	0	0	0	0	0
REQ_MUS_005	0	0	0	0	0
REQ_MUS_006	0	0	0	0	0
REQ_MUS_007	0	0	0	0	0
REQ_PAY_001	0	0	0	0	0
REQ_PAY_002	0	0	0	0	0
REQ_TOR_001	0	0	0	0	0
REQ_TOR_002	0	0	0	0	0
REQ_TOR_003	1	1*	1*	1	1

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_BDS_001	o	o	o	o	o
REQ_BDS_002	o	o	o	o	o
REQ_BDS_003	o	o	o	o	o
REQ_BDS_004	o	o	o	o	o
REQ_BDS_005	o	o	o	o	o
REQ_BDS_006	o	o	o	o	o
REQ_BDS_007	o	o	o	o	o
REQ_PHN_001	o	o	o	o	o
REQ_USR_001	o	o	o	o	o
REQ_USR_002	o	o	o	o	o
REQ_USR_003	o	o	o	o	o
REQ_USR_004	o	o	o	o	o
REQ_USR_005	o	o	o	o	o
REQ_USR_006	o	o	o	o	o
REQ_USR_007	o	o	o	o	o
REQ_RES_001	o	o	o	o	o
REQ_RES_002	o	o	o	o	o
REQ_RES_003	o	o	o	o	o
REQ_RES_004	o	o	o	o	o
REQ_RES_008	o	o	o	o	o
REQ_RES_009	o	o	o	o	o
REQ_MAP_001	o	o	o	o	o
REQ_MAP_002	o	o	o	o	o
REQ_MAP_004	o	o	o	o	o
REQ_MAP_005	o	o	o	o	o
REQ_MAP_006	o	o	o	o	o
REQ_MAP_007	o	o	o	o	o
REQ_MAP_008	o	o	o	o	o
REQ_MAP_009	ı	o	o	o	o
REQ_NAV_001	o	o	o	o	o
REQ_NAV_002	o	o	o	o	o
REQ_NAV_003	o	o	o	o	o
REQ_NAV_004	o	o	o	o	o
REQ_NOT_001	o	o	o	o	o
REQ_NOT_002	o	o	o	o	o

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_NOT_003	0	0	0	0	0
REQ_NOT_004	0	0	0	0	0
REQ_NOT_006	0	0	0	0	0
REQ_NOT_007	0	0	0	0	0
REQ_NOT_009	0	0	0	0	0
REQ_NOT_010	0	0	0	0	0
REQ_LGN_001	0	0	0	0	0
REQ_LGN_002	0	0	0	0	0
REQ_LGN_003	0	0	0	0	0
REQ_SPM_001	1	1	1	0	1
REQ_SPM_002	1	1	1	1	1
REQ_SPM_003	1	1	1	1	1
REQ_SPM_004	1	1	1	1	1
REQ_SPM_005	0	1	0	0	0
REQ_SPM_006	0	1	1	1*	1
REQ_SPM_007	1	1	1	1	1
REQ_NF_001	0	0	0	1	0
REQ_NF_002	0	0	0	0	0
Impacted set size	8	11	9	9	9

D.4. Task 3 (REQ_MAP_002)

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_SCH_001	0	0	0	0	0
REQ_WBS_001	0	0	0	0	0
REQ_WBS_002	0	0	0	0	0
REQ_WBS_003	0	0	0	0	0
REQ_WBS_004	0	0	0	0	0
REQ_WBS_005	0	0	0	0	0
REQ_MUS_001	0	0	0	0	0
REQ_MUS_002	0	0	0	0	0
REQ_MUS_003	0	0	0	0	0
REQ_MUS_004	0	0	0	0	0
REQ_MUS_005	0	0	0	0	0
REQ_MUS_006	0	0	0	0	0

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_MUS_007	o	o	o	o	o
REQ_PAY_001	o	o	o	o	o
REQ_PAY_002	o	o	o	o	o
REQ_TOR_001	o	o	o	o	o
REQ_TOR_002	o	o	o	o	o
REQ_TOR_003	o	o	o	o	o
REQ_BDS_001	o	o	o	o	o
REQ_BDS_002	o	o	o	o	o
REQ_BDS_003	o	o	o	o	o
REQ_BDS_004	o	o	o	o	o
REQ_BDS_005	o	o	o	o	o
REQ_BDS_006	o	o	o	o	o
REQ_BDS_007	o	o	o	o	o
REQ_PHN_001	o	o	o	o	o
REQ_USR_001	o	o	o	o	o
REQ_USR_002	o	o	o	o	o
REQ_USR_003	o	o	o	o	o
REQ_USR_004	o	o	o	o	o
REQ_USR_005	o	o	o	o	o
REQ_USR_006	o	o	o	o	o
REQ_USR_007	o	o	o	o	o
REQ_RES_001	o	o	o	o	o
REQ_RES_002	o	o	o	o	o
REQ_RES_003	o	o	o	o	o
REQ_RES_004	o	o	o	o	o
REQ_RES_008	o	o	o	o	o
REQ_RES_009	o	o	o	o	o
REQ_MAP_001	o	o	o	o	o
REQ_MAP_002	I	I	I	I	I
REQ_MAP_004	o	o	o	o	o
REQ_MAP_005	o	o	o	o	o
REQ_MAP_006	o	o	o	o	o
REQ_MAP_007	o	o	o	o	o
REQ_MAP_008	o	o	o	o	o
REQ_MAP_009	o	o	o	o	o

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_NAV_001	o	o	o	o	o
REQ_NAV_002	o	o	o	o	o
REQ_NAV_003	o	o	o	o	o
REQ_NAV_004	o	o	o	o	o
REQ_NOT_001	o	o	o	o	o
REQ_NOT_002	o	o	o	o	o
REQ_NOT_003	o	o	o	o	o
REQ_NOT_004	o	o	o	o	o
REQ_NOT_006	o	o	o	o	o
REQ_NOT_007	o	o	o	o	o
REQ_NOT_009	o	o	o	o	o
REQ_NOT_010	o	o	o	o	o
REQ_LGN_001	o	o	o	o	o
REQ_LGN_002	o	o	o	o	o
REQ_LGN_003	o	o	o	o	o
REQ_SPM_001	o	o	o	o	o
REQ_SPM_002	o	o	o	o	o
REQ_SPM_003	o	o	o	o	o
REQ_SPM_004	o	o	o	o	o
REQ_SPM_005	o	o	o	o	o
REQ_SPM_006	o	o	o	o	o
REQ_SPM_007	o	o	o	o	o
REQ_NF_001	o	o	o	o	o
REQ_NF_002	o	o	o	o	o
Impacted set size	1	1	1	1	1

D.5. Task 4 (REQ_NAV_003)

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_SCH_001	o	o	o	o	o
REQ_WBS_001	o	o	o	o	o
REQ_WBS_002	o	o	o	o	o
REQ_WBS_003	o	o	o	o	o
REQ_WBS_004	o	o	o	o	o
REQ_WBS_005	o	o	o	o	o

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_MUS_001	o	o	o	o	o
REQ_MUS_002	o	o	o	o	o
REQ_MUS_003	o	o	o	o	o
REQ_MUS_004	o	o	o	o	o
REQ_MUS_005	o	o	o	o	o
REQ_MUS_006	o	o	o	o	o
REQ_MUS_007	o	o	o	o	o
REQ_PAY_001	o	o	o	o	o
REQ_PAY_002	o	o	o	o	o
REQ_TOR_001	o	o	o	o	o
REQ_TOR_002	o	o	o	o	o
REQ_TOR_003	o	o	o	o	o
REQ_BDS_001	o	o	o	o	o
REQ_BDS_002	o	o	o	o	o
REQ_BDS_003	o	o	o	o	o
REQ_BDS_004	o	o	o	o	o
REQ_BDS_005	o	o	o	o	o
REQ_BDS_006	o	o	o	o	o
REQ_BDS_007	o	o	o	o	o
REQ_PHN_001	o	o	o	o	o
REQ_USR_001	o	o	o	o	o
REQ_USR_002	o	o	o	o	o
REQ_USR_003	o	o	o	o	o
REQ_USR_004	o	o	o	o	o
REQ_USR_005	o	o	o	o	o
REQ_USR_006	o	o	o	o	o
REQ_USR_007	o	o	o	o	o
REQ_RES_001	o	o	o	o	o
REQ_RES_002	o	o	o	o	o
REQ_RES_003	o	o	o	o	o
REQ_RES_004	o	o	o	o	o
REQ_RES_008	o	o	o	o	o
REQ_RES_009	o	o	o	o	o
REQ_MAP_001	o	o	o	o	o
REQ_MAP_002	o	o	o	o	o

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_MAP_004	o	o	o	o	o
REQ_MAP_005	o	o	o	o	o
REQ_MAP_006	o	o	o	o	o
REQ_MAP_007	o	o	o	o	o
REQ_MAP_008	o	o	o	o	o
REQ_MAP_009	o	o	o	o	o
REQ_NAV_001	o	o	o	o	o
REQ_NAV_002	o	o	o	o	o
REQ_NAV_003	I	I	I	I	I
REQ_NAV_004	o	o	o	o	o
REQ_NOT_001	o	o	o	o	o
REQ_NOT_002	o	o	o	o	o
REQ_NOT_003	o	o	o	o	o
REQ_NOT_004	o	o	o	o	o
REQ_NOT_006	o	o	o	o	o
REQ_NOT_007	o	o	o	o	o
REQ_NOT_009	o	o	o	o	o
REQ_NOT_010	o	o	o	o	o
REQ_LGN_001	o	o	o	o	o
REQ_LGN_002	o	o	o	o	o
REQ_LGN_003	o	o	o	o	o
REQ_SPM_001	o	o	o	o	o
REQ_SPM_002	o	o	o	o	o
REQ_SPM_003	o	o	o	o	o
REQ_SPM_004	o	o	o	o	o
REQ_SPM_005	o	o	o	o	o
REQ_SPM_006	o	o	o	o	o
REQ_SPM_007	o	o	o	o	o
REQ_NF_001	o	o	o	o	o
REQ_NF_002	o	o	o	o	o
Impact size	I	I	I	I	I

D.6. Task 5 (REQ_TOR_001)

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_SCH_001	o	o	o	o	o
REQ_WBS_001	o	I	o	o	o
REQ_WBS_002	o	I	o	o	o
REQ_WBS_003	o	I	o	o	o
REQ_WBS_004	o	o	o	o	o
REQ_WBS_005	o	o	o	o	o
REQ_MUS_001	o	o	o	o	o
REQ_MUS_002	o	I	o	o	o
REQ_MUS_003	o	o	o	o	o
REQ_MUS_004	o	I	o	o	o
REQ_MUS_005	o	I	o	o	o
REQ_MUS_006	o	o	o	o	o
REQ_MUS_007	o	o	o	o	o
REQ_PAY_001	o	o	o	o	o
REQ_PAY_002	o	o	o	o	o
REQ_TOR_001	I	I	I	I	I
REQ_TOR_002	I	I	I	o	I
REQ_TOR_003	o	I	o	o	o
REQ_BDS_001	o	o	o	o	o
REQ_BDS_002	o	o	o	o	o
REQ_BDS_003	o	o	o	o	o
REQ_BDS_004	o	o	o	o	o
REQ_BDS_005	o	o	o	o	o
REQ_BDS_006	o	o	o	o	o
REQ_BDS_007	o	o	o	o	o
REQ_PHN_001	o	o	o	o	o
REQ_USR_001	o	o	o	o	o
REQ_USR_002	o	o	o	o	o
REQ_USR_003	o	o	o	o	o
REQ_USR_004	o	o	o	o	o
REQ_USR_005	o	o	o	o	o
REQ_USR_006	o	o	o	o	o
REQ_USR_007	o	o	o	o	o

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_RES_001	o	o	o	o	o
REQ_RES_002	o	o	o	o	o
REQ_RES_003	o	o	o	o	o
REQ_RES_004	o	o	o	o	o
REQ_RES_008	o	o	o	o	o
REQ_RES_009	o	I	o	o	o
REQ_MAP_001	o	o	o	o	o
REQ_MAP_002	o	o	o	o	o
REQ_MAP_004	I	o	o	o	o
REQ_MAP_005	I	o	o	o	o
REQ_MAP_006	o	o	o	o	o
REQ_MAP_007	o	o	o	o	o
REQ_MAP_008	o	o	o	o	o
REQ_MAP_009	I	I	I	o	I
REQ_NAV_001	o	o	o	o	o
REQ_NAV_002	o	o	o	o	o
REQ_NAV_003	I	o	o	o	o
REQ_NAV_004	o	o	o	o	o
REQ_NOT_001	o	o	o	o	o
REQ_NOT_002	o	o	o	o	o
REQ_NOT_003	o	o	o	o	o
REQ_NOT_004	o	o	o	o	o
REQ_NOT_006	o	o	o	o	o
REQ_NOT_007	o	o	o	o	o
REQ_NOT_009	o	I	o	o	o
REQ_NOT_010	o	o	o	o	o
REQ_LGN_001	o	o	o	o	o
REQ_LGN_002	o	o	o	o	o
REQ_LGN_003	o	o	o	o	o
REQ_SPM_001	I	I	I	o	I
REQ_SPM_002	I	I	I	o	I
REQ_SPM_003	o	o	o	o	o
REQ_SPM_004	I	I	I	o	I
REQ_SPM_005	o	I	o	o	o
REQ_SPM_006	o	o	o	o	o

Requirement	Expert	PhD student	Post-doc	MSc student	Mode
REQ_SPM_007	0	0	0	0	0
REQ_NF_001	0	0	0	0	0
REQ_NF_002	0	0	0	0	0
Impacted set size	9	16	6	1	6

E. Box plots

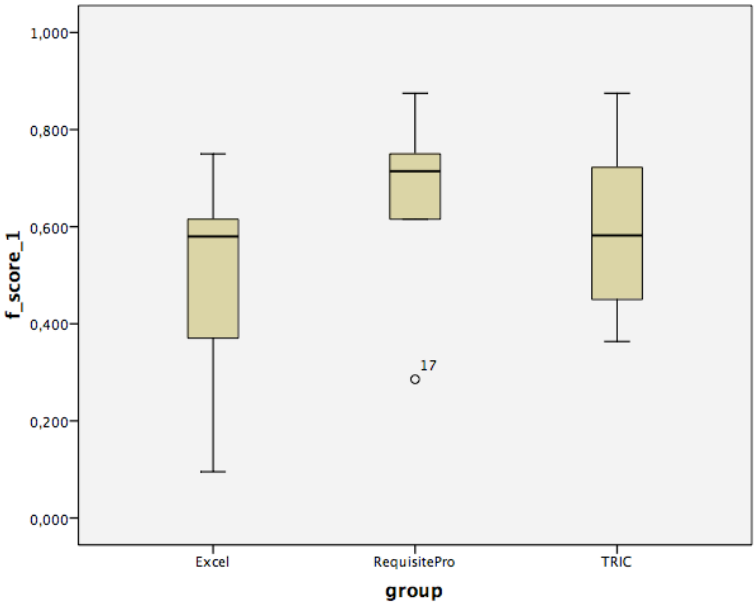
E.1. Introduction

This appendix contains box plots of the *F*-score of change impact predictions and the time taken to complete them in seconds, per group per task.

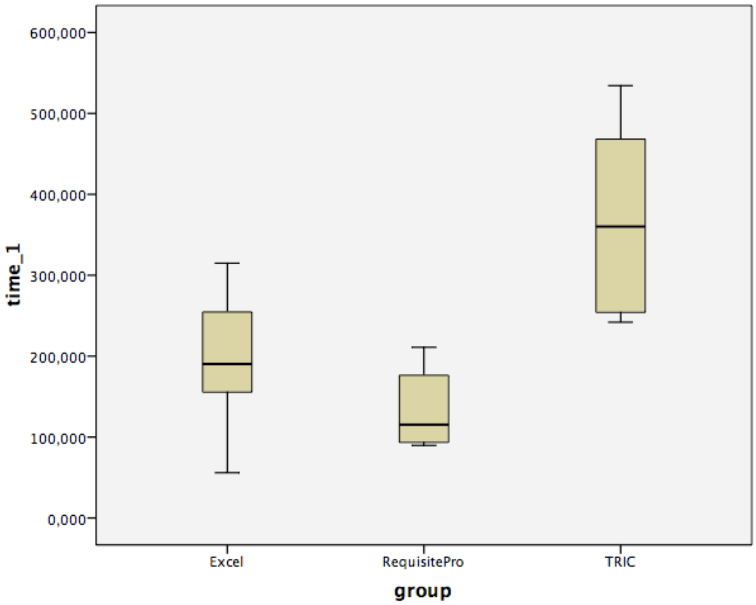
Circles with numbers are cases that were classified as outliers by SPSS.

E.2. Task 1 (REQ_PHN_001)

F-score

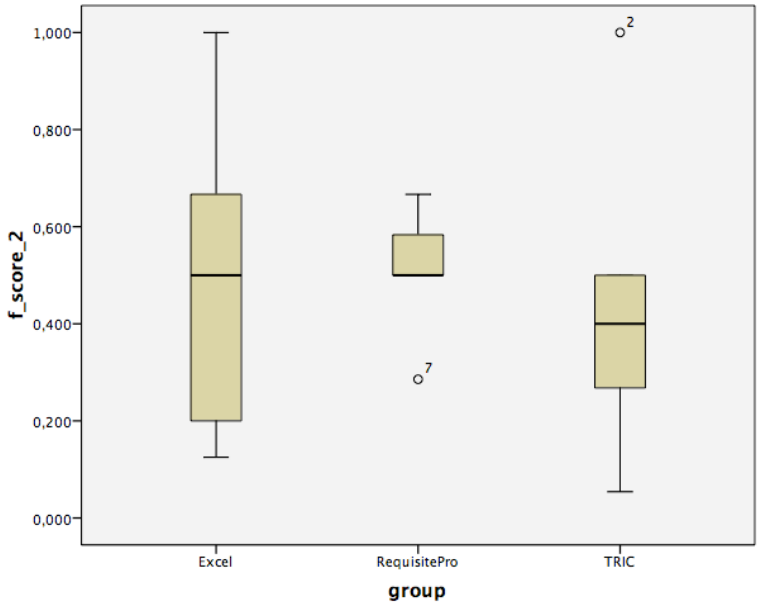


Time

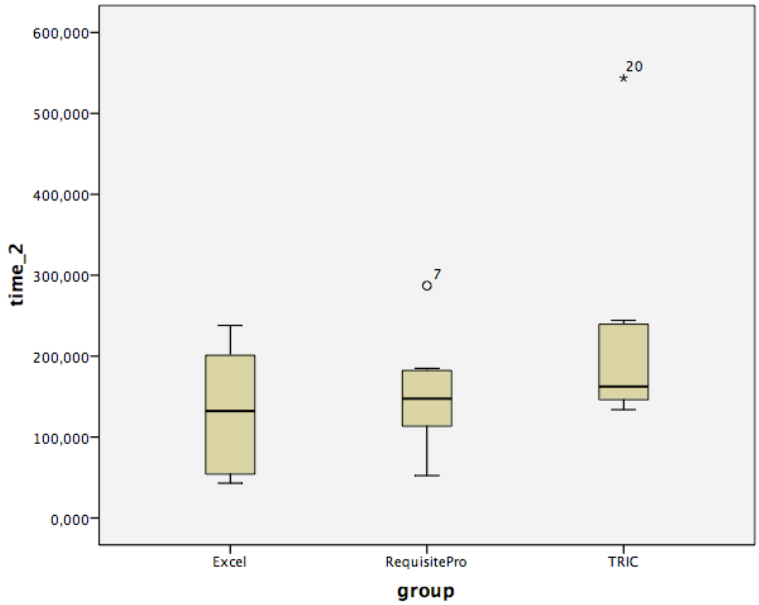


E.3. Task 2 (REQ_SPM_004)

F-score

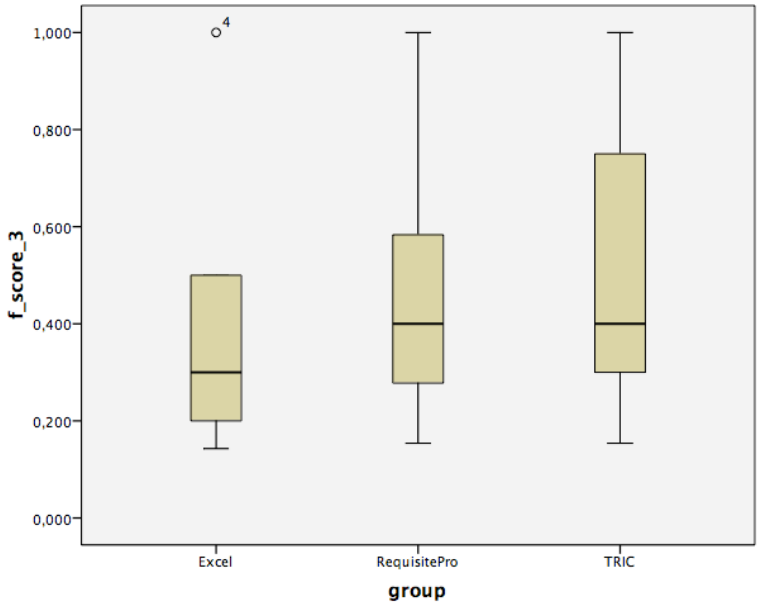


Time

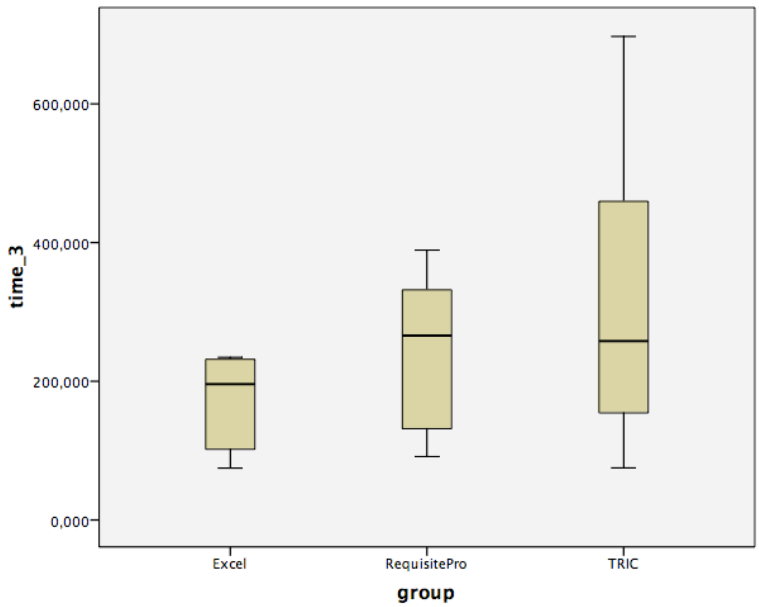


E.4. Task 3 (REQ_MAP_002)

F-score

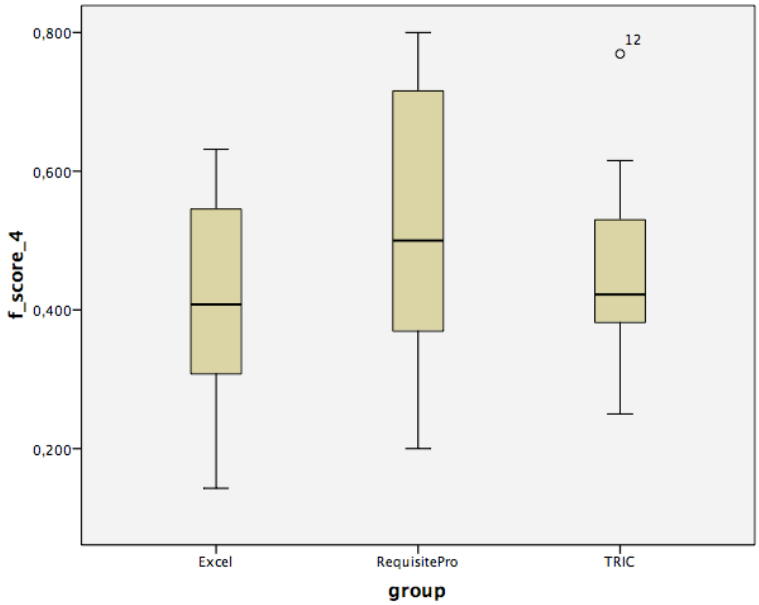


Time

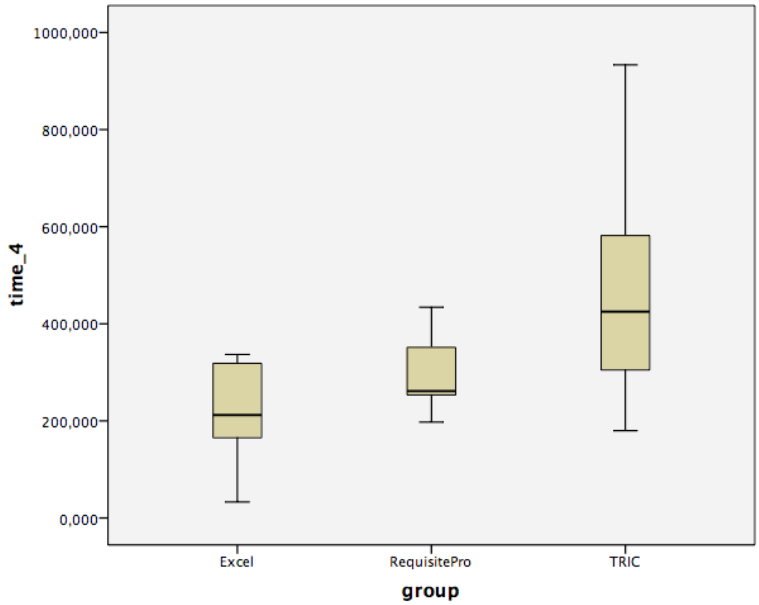


E.5. Task 4 (REQ_NAV_003)

F-score

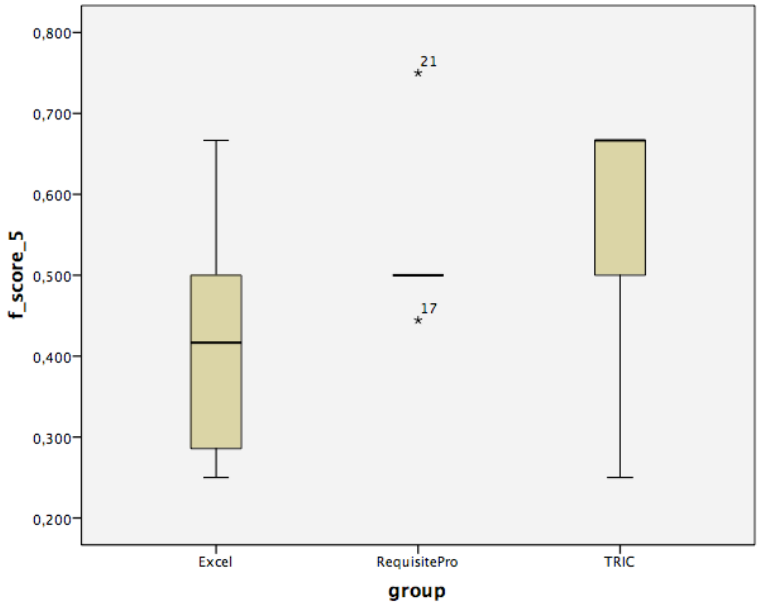


Time

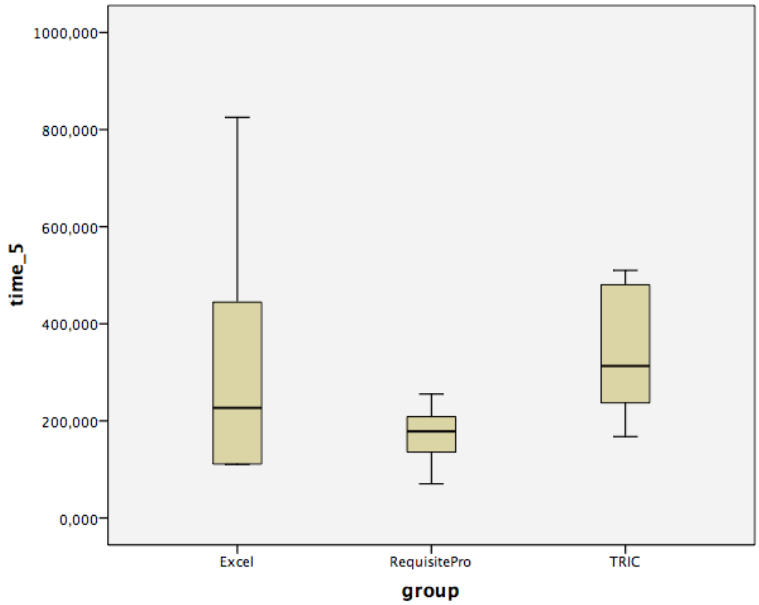


E.6. Task 5 (REQ_TOR_001)

F-score



Time



F. Precision-Recall and ROC graphs

F.1. Introduction

This appendix contains the Precision-Recall graphs and Receiver Operating Characteristics of the change impact predictions by the experimental groups.

F.2. Legend

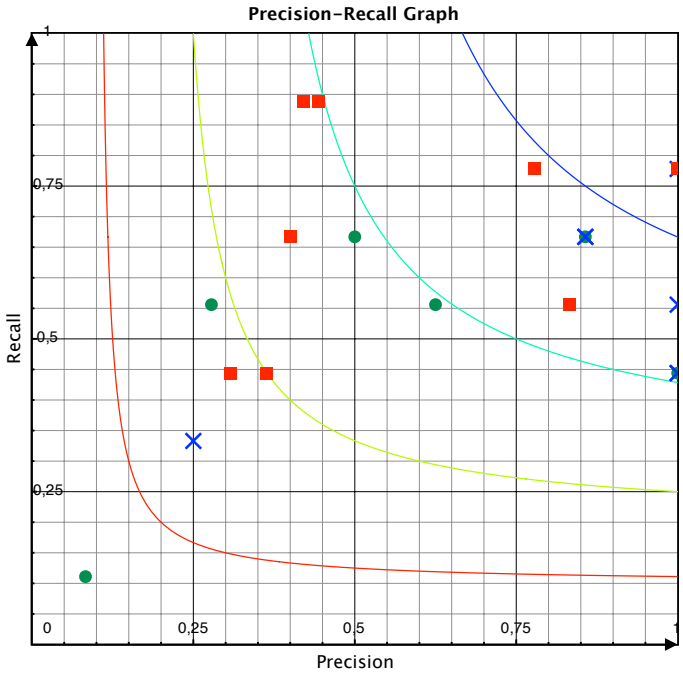
The legend to these graphs is as follows:

Marker	Icon	Group
Circle	•	Microsoft Excel
Cross	×	IBM Rational RequisitePro
Square	▪	TRIC

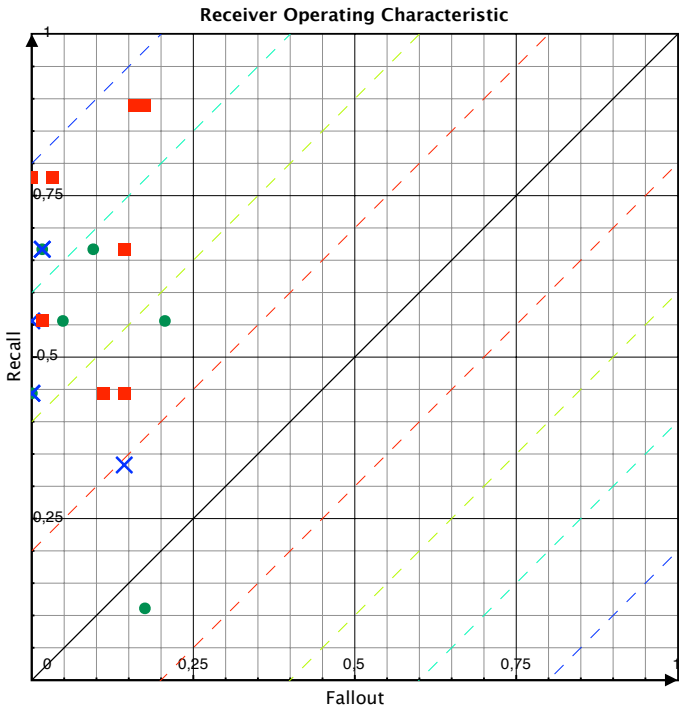
A statistical analysis follows this graphical presentation to more precisely discover differences between the performance of groups using Microsoft Excel, IBM Rational RequisitePro and TRIC.

F.3. Task 1

Precision-Recall graph

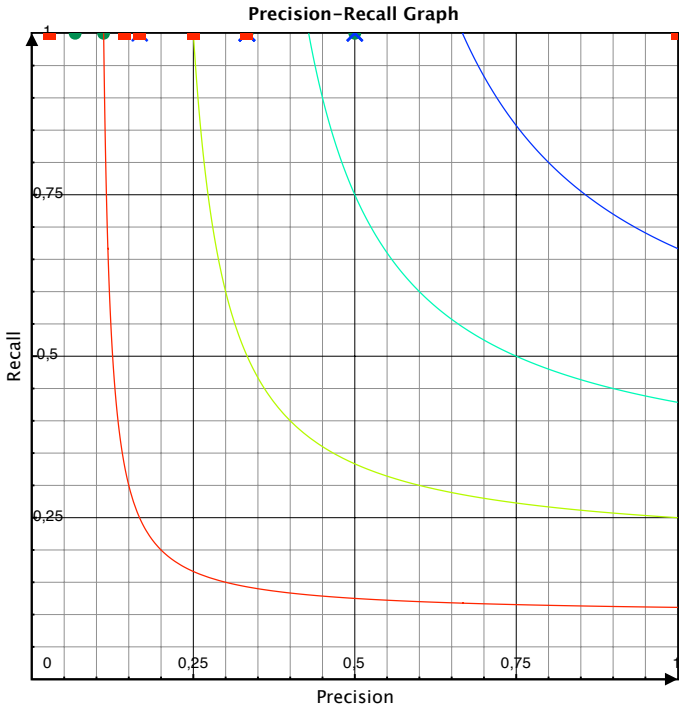


Receiver Operating Characteristic

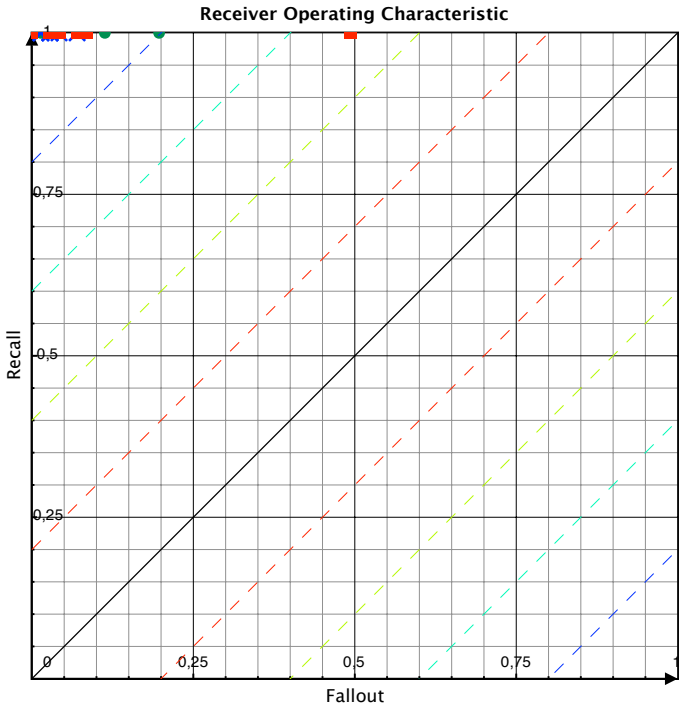


F.4. Task 2

Precision-Recall graph

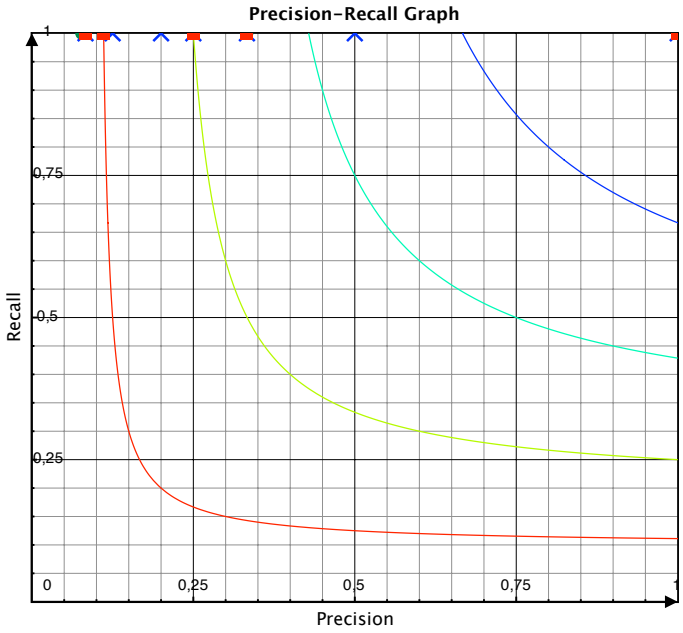


Receiver Operating Characteristic

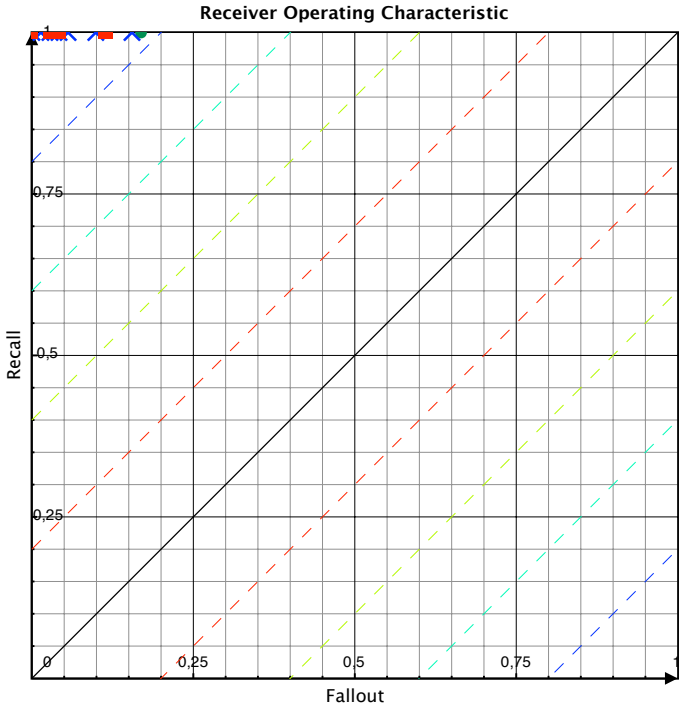


F.5. Task 3

Precision-Recall graph

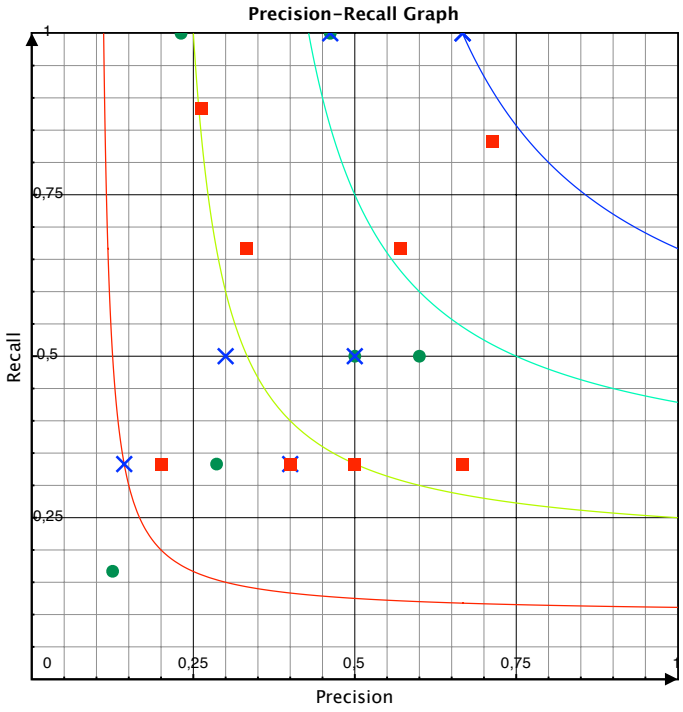


Receiver Operating Characteristic

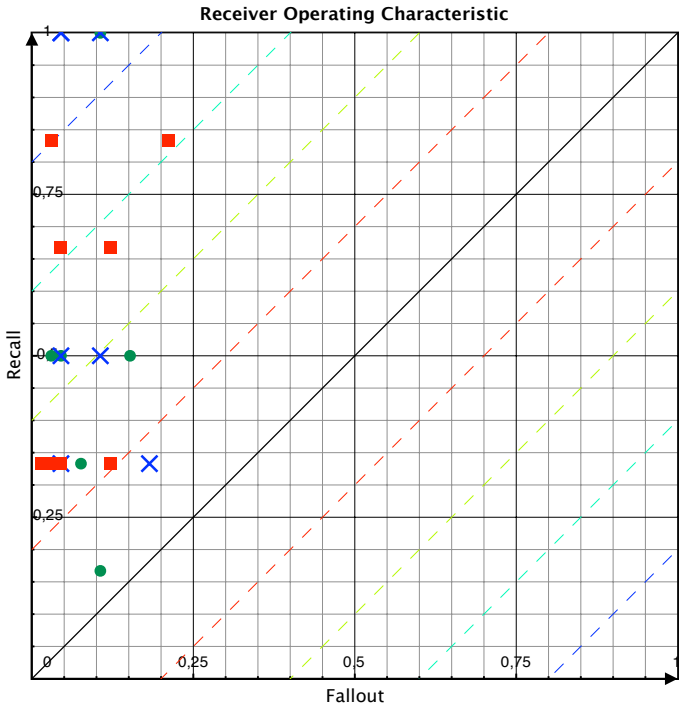


F.6. Task 4

Precision-Recall graph

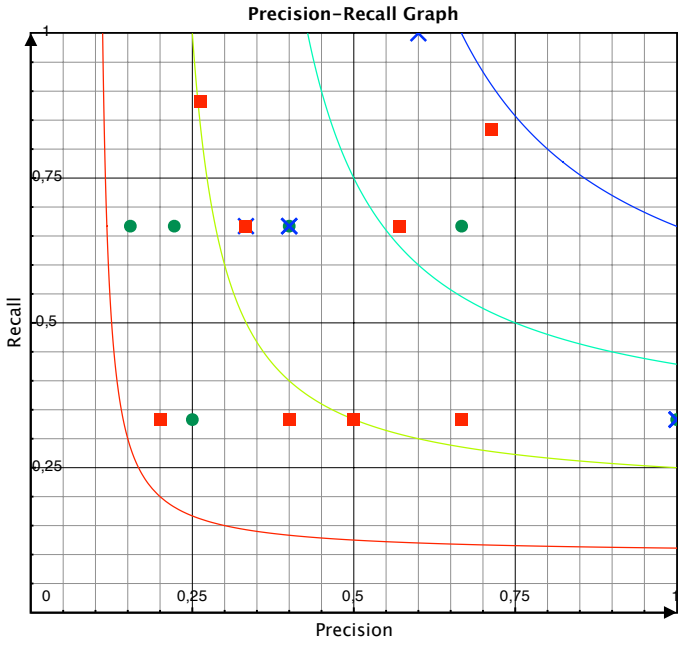


Receiver Operating Characteristic

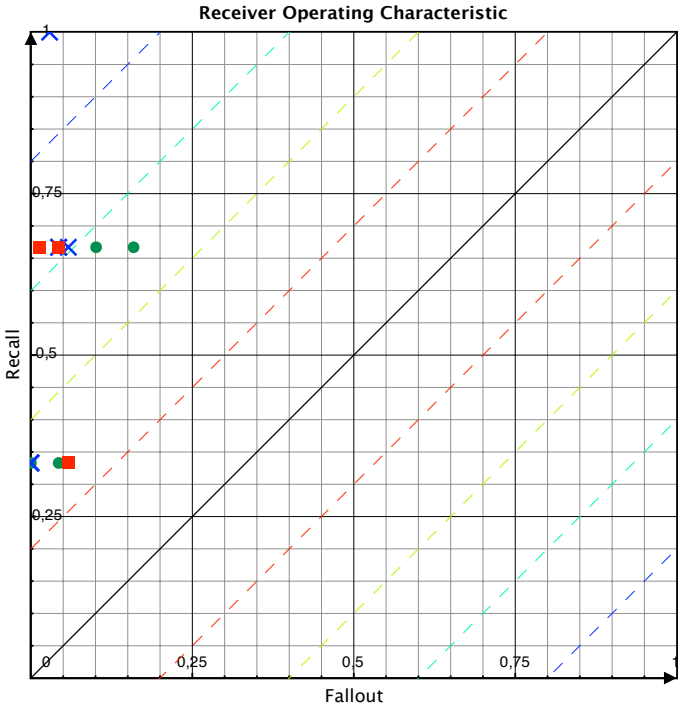


F.7. Task 5

Precision-Recall graph



Receiver Operating Characteristic



G. WASP requirements

G.1. Introduction

The following pages are copied from the WASP requirements specification [18].

4 Requirements

This chapter presents the requirements for the WASP platform and some of the demo WASP applications. Because the actual distinction between the platform and the applications is a design decision, the requirements have not been grouped using this distinction; instead they have been grouped using the same logical grouping as used for the use cases. It is up to the designers of the WASP platform to decide if a requested functionality is generic enough to become part of the platform or if it should be left to the WASP application. However, in order to make the requirements better understandable, they are assigned in their definitions to either the platform or an application. The requirements also provide several wishes for functionality in the 3G platform from the point of view of the WASP platform. The requirements have been distilled from the use cases presented in the previous chapter.

Traditional classification divides requirements into two groups: functional requirements that describe the behavior that the system should provide to the environment, and non-functional requirements that describe how this functionality should be provided, with respect to the performance, control capabilities, economic aspects, or efficiency of the system. Section 4.1 lists the functional requirements derived from the use cases, while section 4.2 mentions some non-functional requirements.

4.1 Functional requirements

4.1.1 Scheduling

Requirement ID: REQ_SCH_001	**Category**: Functional	**From use case**: UC_SCH_001
There SHOULD be an application that provides functionality for mobile users to propose and schedule meetings.		
Description:		
That application enables mobile users to make mutual appointments via different channels including • E-mail (free-format) • SMS • Agenda's (e.g. Outlook) depending on the status and technical capabilities of the invited users.		

4.1.2 Web services

Requirement ID: REQ_WBS_001	**Category**: Functional	**From use case**: UC_WBS_001 UC_WBS_002
The WASP platform SHALL provide functionality to find services that match the user's explicit need and obey the restrictions following from the user's profile and current context. For the found services the WASP platform SHALL provide (links to) the service profiles.		
Description:		
These services include • Physical services (visiting possibilities) • User services (telephone, e-mail, web-site) • Application services (web services)		

Requirement ID: REQ_WBS_002	**Category**: Functional	**From use case**: UC_WBS_001 UC_TOR_001
The WASP platform MUST be able to collect profile and context information on the user from multiple sources.		
Description:		
These sources include • Internal (profiles, user history, …); • External (context, …); • Mobile network (user status, location, via interfaces on the 3G platform, see REQ_USS_001 and REQ_USL_001 in [Laa2002]).		

Requirement ID: REQ_WBS_003	**Category**: Functional	**From use case**: UC_WBS_001 UC_TOR_001
The WASP platform MUST be able to combine knowledge about the end-user (in terms of profile and context) and a specific information request into a search query on the service registry.		
Description:		
The WASP platform has logical intelligence to restrict the registry search given profile and context information.		

Requirement ID: REQ_WBS_004	**Category**: Functional	**From use case**: UC_WBS_001 UC_WBS_002
The tourist application MUST support invocation of user-selected services based on the service profile.		
Description:		
For web services, the web service client can be pre-fabricated as well as dynamically generated, for email services it provides a hook to a relevant email application, for telephone services it offers call-setup, and for web sites links shall be provided.		

Requirement ID: REQ_WBS_005	**Category**: Functional	**From use case**: UC_TOR_001 UC_WBS_001 UC_WBS_002
The tourist application SHALL be able to represent the results from a registry search or service interaction in an orderly manner.		
Description:		
In terms of lists and maps with interaction possibilities for the end-user		

4.1.3 Museum

Requirement ID: REQ_MUS_001	**Category**: Functional	**From use case**: UC_MUS_001
A museum application MAY offer a personalized catalogue service.		
Description:		
Such a catalogue service presents the user with a (if requested by the user a personalized) overview of the museum's collection.		

Requirement ID: REQ_MUS_002	**Category**: Functional	**From use case**: UC_MUS_002
A museum application MAY offer a tour reservation service.		
Description:		
A user is able to reserve a tour through the museum with this service for one or more people for a specific date and time.		

Requirement ID: REQ_MUS_003	**Category**: Functional	**From use case**: UC_MUS_003
A museum application MAY offer a translated guide service.		
Description:		
When the museum offers a guide for the museum, such as the catalogue, it would be nice if it can translate the guide to different languages for people of different nationalities.		

Requirement ID: REQ_MUS_004	**Category**: Functional	**From use case**: UC_MUS_001 UC_MUS_003 UC_RES_003
The WASP platform MUST provide services that MAY be used by WASP applications in order to personalize content for a specific end-user based on his or her user profile.		
Description:		
This can be done by providing the service with elements from the user's profile and context, or by post-processing the response of the service given the user's profile and context. The post-processing can be done by the WASP application itself, where necessary using personalization services provided by the platform, or referred to another 3rd party service provider, e.g. translation services. Examples are the personalization of a museum-catalogue or the personalization of a menu in a restaurant.		

Requirement ID: REQ_MUS_005	**Category**: Functional	**From use case**: UC_MUS_001
The WASP platform SHOULD provide services that MAY be used by WASP applications in order to personalize content for a group of end-users based on their user profiles.		
Description:		
With group personalization the focus is to personalize content not for one specific user, but for a group of users, taking into account each individual's preferences and interests. Personalization for groups can either be done by combining the individual user profiles and treat the group profile the same as an individual profile for personalization, or by first personalizing for each individual user and than combining the results into a group result.		

Requirement ID: REQ_MUS_006	**Category**: Functional	**From use case**: UC_MUS_002 UC_PAY_001 UC_PAY_002
The WASP platform must provide services that MAY be used by a WASP application to charge the user for using one of his services.		
Description:		
A real time confirmation of the money issuer is required to complete the transaction.		

Requirement ID: REQ_MUS_007	**Category**: Functional	**From use case**: UC_MUS_004
The WASP platform MUST allow end-users to provide profile and context information explicitly to applications or the platform.		
Description:		
In the following cases: • At the first interaction moment, to bootstrap the personalization • When no profile information is available • When profile information needs to be overruled		

4.1.4 Payment

Requirement ID: REQ_PAY_001	**Category**: Functional	**From use case**: UC_PAY_001 UC_PAY_002
A user's profile MUST contain at least one charging point for a specific user.		
Description:		
Different users have different ways for payment, called charging points, and also different financial parties (banks, mobile operator, etc) are involved, each with their own way of working. Via the user' profile a service can determine the possible charging points for a user and his preferred charging point.		

Requirement ID: REQ_PAY_002	**Category**: Functional	**From use case**: UC_PAY_001 UC_PAY_002
A WASP application MUST be authorized by the end-user before charging his account.		
Description:		
This can be done by delegation, via the profile or in real time.		

4.1.5 Tourist service

Requirement ID: REQ_TOR_001	**Category**: Functional	**From use case**: UC_TOR_001
The WASP platform SHALL provide functionality to find points of interest that match the user's explicit need and obey the restrictions following from the user's profile and current context.		
Description:		
Points of interest can be found in the following ways (or combinations of those): • Find points of interest within a specific area • Find points of interest that match with the profiles of a users or a group of users • Find points of interest that match an explicit need (specified in free text, categorizations, or identifiers)		

Requirement ID: REQ_TOR_002	**Category**: Functional	**From use case**: UC_TOR_001
The WASP platform MUST allow end-users to place triggers to be alerted automatically for a certain type of point of interests, based on his/her context.		
Description:		
This way, end-users can be warned when for example: • new points of interest become available; • the user comes nearby a point of interest; • an existing point of interest is updated.		

Requirement ID: REQ_TOR_003	**Category**: Functional	**From use case**: UC_TOR_001
The WASP platform MUST allow end-users to place triggers in the platform to be alerted automatically for a certain type of services and events, based on his/her context.		
Description:		
This way, end-users can be warned when for example: • new services or events become available; • the user comes close to a point of interest that provides a certain service or has a certain event; • an existing service of event is updated.		

4.1.6 Buddies

Requirement ID: REQ_BDS_001	**Category**: Functional	**From use case**: UC_BDS_001
The WASP platform SHALL provide functionality to search for other people registered in the platform.		
Description:		
Users must be able to search for other people that are registered.		

Requirement ID: REQ_BDS_002	**Category**: Functional	**From use case**: UC_BDS_001
The WASP platform SHALL allow end-users to maintain a buddy list.		
Description:		
The end-user must be able to add and remove buddies (other WASP users) from his buddy list.		

Requirement ID: REQ_BDS_003	**Category**: Functional	**From use case**: UC_BDS_001
The WASP platform SHALL provide functionality for end-users to communicate with their buddies.		
Description:		
Users must for example be able to phone their buddies, chat with them or send them messages.		

Requirement ID: REQ_BDS_004	**Category**: Functional	**From use case**: UC_BDS_001
The WASP platform SHOULD be able to use several communication services offered by the 3G platform.		
Description:		
E.g. chatting, phone conversations, message exchange etc. (see REQ_CC_004, REQ_CC_005 and REQ_MSG_001 in [Laa2002]).		

Requirement ID: REQ_BDS_005	**Category**: Functional	**From use case**: UC_BDS_002
The WASP platform SHALL be able to update status and location information of buddies in the user's buddy list based either on request of the user, after login into the WASP platform and at a regular interval.		
Description:		
In the user's buddy list, information is displayed concerning the user's status and current location (if that buddy allows this information to be used). This information is updated at the moment the user logs in, the user manually requests an update of this information, or automatically at a regular interval as long as the user is logged in. This information is requested from the WASP platform.		

Requirement ID: REQ_BDS_006	**Category**: Functional	**From use case**: UC_BDS_002
The WASP platform MUST allow end-users to set alerts in the platform on changes in the status and/or location of his buddies.		
Description:		
By setting such alerts, a user can specify that he for example is notified when a specific buddy comes online or within a certain distance of himself.		

Requirement ID: REQ_BDS_007	**Category**: Functional	**From use case**: UC_BDS_002
When changes are discovered in the status and/or location of a user's buddy, the WASP platform MUST sent out notifications according to the alerts set by the user (see also REQ_NOT_006).		
Description:		
When a change in a buddies status and/or location means that an alert is triggered, the specified notifications must be sent to the user.		

4.1.7 Phone Call

Requirement ID: REQ_PHN_001	**Category**: Functional	**From use case**: UC_PHN_001
A WASP application SHALL be able to setup a phone call connection using the 3G Platform via the WASP platform.		
Description:		
The 3G Platform requirements for setting up phone calls are REQ_CC_004 and REQ_CC_005 in [Laa2002].		

4.1.8 User Management

Requirement ID: REQ_USR_001	**Category**: Functional	**From use case**: UC_USR_001 UC_USR_004 UC_RES_003 UC_NOT_003
The WASP platform SHALL allow status information to be stored, updated, retrieved and deleted in the user's profile in the WASP platform if the user allows this.		
Description:		
The 3G Platform, the WASP Platform and WASP applications can store status information in the profile of the user, of course taking into account security rights via the WASP platform.		

Requirement ID: REQ_USR_002	**Category**: Functional	**From use case**: UC_USR_002 UC_RES_003
The WASP platform SHALL allow end-users to give and remove Create, Read, Update, Delete (CRUD) access rights on parts of the user profile to other users, specific WASP applications and the 3G Platform.		
Description:		
End-users can allow other users, WASP applications and/or the 3G Platform to access parts of their user profile. End-users can give different access rights: create (insert), read, update and/or delete. Access rights given on the combination of per user per application basis.		

Requirement ID: REQ_USR_003	**Category**: Functional	**From use case**: UC_USR_002
The WASP platform SHALL allow other users, WASP applications and/or the 3G Platform to ask the end-user for access rights on parts of the user profile.		
Description:		
If another user, WASP application or the 3G Platform wants to access part of a user profile that it doesn't have the access privileges for, it must be possible that the user is ask whether or not he wants to allows this access (no, one-time or permanently).		

Requirement ID: REQ_USR_004	**Category**: Functional	**From use case**: UC_USR_003
The WASP platform SHOULD allow end-users to delegate and remove the rights to give access (see REQ_USR_001) to parts of the user profile to other users.		
Description:		
End-users can allow other users to give access privileges to WASP applications or the 3G Platform or other users.		

Requirement ID: REQ_USR_005	**Category**: Functional	**From use case**: UC_USR_005
The WASP platform SHALL provide functionality to retrieve the current location of the user.		
Description:		
Taking into account access rights, it should be possible for WASP applications, the 3G Platform and other users to retrieve the current location of the user.		

Requirement ID: REQ_USR_006	**Category**: Functional	**From use case**: UC_USR_004
The WASP platform MUST be able to retrieve status information from the 3G Platform.		
Description:		
Some status information can be requested from the 3G Platform, such as online/offline status and on the phone status (see REQ_US_001 in [Laa2002]).		

Requirement ID: REQ_USR_007	**Category**: Functional	**From use case**: UC_USR_004 UC_MAP_001
The WASP platform MUST be able to retrieve location information from the 3G Platform.		
Description:		
The 3G Platform is able to provide the current geographic location of the user, providing the user is online (see REQ_USL_001 in [Laa2002]).		

4.1.9 Restaurant

Requirement ID: REQ_RES_001	**Category**: Functional	**From use case**: UC_RES_001
A restaurant application MAY provide a table reservation service.		
Description:		
End-users may be able to reserve a table at a restaurant that is known to the WASP platform using an online service provided by that restaurant.		

Requirement ID: REQ_RES_002	**Category**: Functional	**From use case**: UC_RES_001
A restaurant application MAY provide a service that allows users to send out invitations to other dinner guests after a table has been reserved.		
Description:		
When an end-user has reserved a table at a restaurant, the restaurant may offer the end-user the possibility to automatically send out a dinner invitation to the other participants of the dinner. The user can then provide the names/addresses of those users and a personal message.		

Requirement ID: REQ_RES_003	**Category**: Functional	**From use case**: UC_RES_001
It MUST be possible for a WASP application to send out messages to users using the WASP platform.		
Description:		
The WASP platform must be facilities that allow WASP applications to send messages to users.		

Requirement ID: REQ_RES_004	**Category**: Functional	**From use case**: UC_RES_002
A restaurant application MAY provide a service that allows users to see photographs of the interior of the restaurant.		
Description:		
End-users may be able to see how the restaurant looks like (in order to get a feeling of the restaurant) by looking at photographs of the restaurant.		

Requirement ID: REQ_RES_005	**Category**: Functional	**From use case**: UC_RES_002
A WASP application SHOULD be able to automatically adapt content and the presentation of functionality to the user's device, for which the WASP platform MAY offer some generic adaptation services.		
Description:		
Because different user's have different devices, with different screen properties (size, number of colors etc.) it should be possible that applications adapt their content and the presentation of the functionality to the characteristics of the user's device.		

Requirement ID: REQ_RES_008	**Category**: Functional	**From use case**: UC_RES_003
A restaurant application SHOULD allow end-users to give feedback on items from a menu.		
Description:		
Such feedback represents the actual opinion of the user about that specific dish, which can be used to better learn the tastes and preferences of the user in food.		

Requirement ID: REQ_RES_009	**Category**: Functional	**From use case**: UC_RES_003
A restaurant application SHOULD be able to learn from feedback provided by the user and store the newly learn tastes and preferences of the user in the user's profile.		
Description:		
Information learned from the users should be able to be stored in the user's profile (see also REQ_RES_007).		

4.1.10 Map

Requirement ID: REQ_MAP_001	**Category**: Functional	**From use case**: UC_MAP_001
The WASP platform MUST be able to draw a map of a given area.		
Description:		
It MUST be possible to provide the user with a digital map of a given area.		

Requirement ID: REQ_MAP_002	**Category**: Functional	**From use case**: UC_MAP_001
The WASP platform MUST be able to show the location of users on a map.		
Description:		
It MUST be possible to show the current location of the end-user and/or (a number of) his/her buddies on the map.		

Requirement ID: REQ_MAP_004	**Category**: Functional	**From use case**: UC_MAP_001
The WASP platform MUST be able to show the location of various points of interests on a map.		
Description:		
It MUST be possible to draw a map of a given area that displays the geographic location of various points of interests.		

Requirement ID: REQ_MAP_005	**Category**: Functional	**From use case**: UC_MAP_001
The WASP platform SHALL be able to obtain the geographic location of points of interests.		
Description:		
Requirement REQ_MAP_004 implies that the platform SHALL be able to obtain the geographic location of points of interest, in order to be able to show them on the map.		

Requirement ID: REQ_MAP_006	**Category**: Functional	**From use case**: UC_MAP_001
The WASP platform MUST be able to display a route connecting two or more points on a map.		
Description:		
It MUST be possible to display a route between the end-user's location and other users or points of interest on a map.		

Requirement ID: REQ_MAP_007	**Category**: Functional	**From use case**: UC_MAP_001
The WASP platform SHOULD be able to provide the end-user with walking or driving instructions.		
Description:		
It SHOULD be possible to provide the end-user with walking or driving instructions that explain to the user how to get from one point to another.		

Requirement ID: REQ_MAP_008	**Category**: Functional	**From use case**: UC_MAP_001
The WASP platform MUST be able to render all maps on small, mobile devices, as well as large, fixed terminals.		
Description:		
It MUST be possible to render all maps on a variety of devices, adapting the size and possibly detail level of the map for easy viewing (that is, no scrolling needed).		

Requirement ID: REQ_MAP_009	**Category**: Functional	**From use case**: UC_MAP_001 UC_WBS_001 UC_WBS_002
The platform MUST be able to access points of interests and services from a map.		
Description:		
It MUST be possible to access information about points of interests and links to the services provided by the points on interest from the map on which they are displayed.		

4.1.11 Personalized Dynamic Navigation

Requirement ID: REQ_NAV_001	**Category**: Functional	**From use case**: UC_NAV_001, UC_NAV_002
The WASP platform SHOULD be able to calculate a route between two or more arbitrarily chosen points, avoiding traffic jams or construction works.		
Description:		
It SHOULD be possible to calculate a route between two ore more arbitrarily chosen points, on an address-by-address basis. Traffic jams or congestion due to construction works SHOULD be avoided.		

Requirement ID: REQ_NAV_002	**Category**: Functional	**From use case**: UC_NAV_001 UC_NAV_002
The WASP platform SHOULD have access to information about traffic jams and construction works, as well as suggested alternative routes.		
Description:		
Implied by requirement REQ_NAV_001.		

Requirement ID: REQ_NAV_003	**Category**: Functional	**From use case**: UC_NAV_001
The WASP platform SHOULD be possible to determine the location of touristic attractions close to a calculated route.		
Description:		
In order to allow for touristic detours, the platform SHOULD be able to determine which touristic attractions are close to a calculated route.		

Requirement ID: REQ_NAV_004	**Category**: Functional	**From use case** :UC_NAV_001 UC_NAV_002
The WASP platform SHOULD be able to determine whether a traffic jam lies on a suggested route.		
Description:		
The platform SHOULD be able to determine whether the end-user is affected by construction works or traffic jams along a suggested route, in order to assure that the suggested route is free of traffic jams or congestions due to construction works.		

4.1.12 Notifications

Requirement ID: REQ_NOT_001	**Category**: Functional	**From use case**: UC_NOT_001
The WASP platform MUST allow end-users to set an alert on an event.		
Description:		
The end-user MUST be able to set an alert on an event, so that, as soon as the event occurs, the end-user is notified about this. Examples of events are those as given REQ_TOR_002, REQ_TOR_003 and REQ_BDS_006.		

Requirement ID: REQ_NOT_002	**Category**: Functional	**From use case**: UC_NOT_001
The WASP platform SHOULD allow the end-user to specify the notification type when setting an alert.		
Description:		
The end-user SHOULD be able to specify the type of notification he/she receives when an event actually occurs (and thus the alert activates). This can be done on a per-alert basis. Examples of notification types are a phone call, a message, a sound played on his mobile device etc.		

Requirement ID: REQ_NOT_003	**Category**: Functional	**From use case**: UC_NOT_001
The WASP platform MUST maintain a list of events the end-user can be notified about.		
Description:		
The user MUST be able to choose from a list of events that are supported by the platform, according to UC_NOT_001, so the platform MUST maintain such a list.		

Requirement ID: REQ_NOT_004	**Category**: Functional	**From use case**: UC_NOT_002
The WASP platform MUST allow the end-user to remove previously set alerts on events		
Description:		
The end-user MUST be able to remove a previously set alert for events, so that the user is no longer notified about the occurrence of the event, should the event occur.		

Requirement ID: REQ_NOT_006	**Category**: Functional	**From use case**: UC_NOT_003
The WASP platform MUST notify the end-user about the occurrence of an event for which an alert was set, as soon as the event occurs.		
Description:		
The end-user MUST be notified about the occurrence of an event, if and only if an alert for that event has been set before, at the moment the event occurs.		

Requirement ID: REQ_NOT_007	**Category**: Functional	**From use case**: UC_NOT_003
The WASP platform SHOULD be able to decide how to notify the user of an alert for which an event was set.		
Description:		
The platform SHOULD be able to decide how to notify the user (e.g., via e-mail, instant message, Short Message Service, telephone call), taking into account the preferred notification mechanism supplied by the user (see REQ_NOT_002) and the user status.		

Requirement ID: REQ_NOT_009	**Category**: Functional	**From use case**: UC_NOT_003
The WASP platform MUST actively monitor all events.		
Description:		
The platform MUST actively monitor all events, in order to notify all users that are interested in the event (i.e., all users who have set an alert for the event) in time. Implied by REQ_NOT_006.		

Requirement ID: REQ_NOT_010	**Category**: Functional	**From use case**: UC_NOT_003
If the user cannot be notified of the event the first time, the WASP platform SHOULD retry to notify the user of the occurrence of the event, until the user has been notified or a specified time-out elapses.		
Description:		
If, for some reason, the user cannot be notified of an event for which an alert was set, the platform SHOULD try to notify the user until the user has been successfully notified or a specified time-out elapses.		

4.1.13 Login

Requirement ID: REQ_LGN_001	**Category**: Functional	**From**: SC_001 item 3 and 4
There MUST not be any application data on the end-user's device. Only terminal capabilities MAY be terminal dependent and those capabilities MAY be stored on the end-user's device.		
Description:		
Replaces a login use case. This is required to enable an end-user to use different terminals.		

Requirement ID: REQ_LGN_002	**Category**: Functional	**From use case**: UC_LGN_001 UC_PAY_002 UC_TOR_001
The WASP platform MUST maintain authentication credentials per user.		
Description:		
In order to authenticate a user you need authentication credentials in order to identify the user uniquely, e.g. user ID and password.		

Requirement ID: REQ_LGN_003	**Category**: Functional	**From use case**: UC_LGN_001
The WASP platform SHOULD be able to request the user's ID from the 3G platform.		
Description:		
In case the user logs in on a mobile device it would be nice if the WASP platform could 'inherit' some of the information that the 3G platform has about the user (see REQ_IPT_001 in [Laa2002]).		

4.1.14 Service Profile Management

Requirement ID: REQ_SPM_001	**Category**: Functional	**From use case**: UC_SPM_001 UC_SPM_002
The WASP platform SHOULD have a (limited) number of classification schemes for POIs and services.		
Description:		
To be able to support the creation of profiles it should be know what type of information is required for the different types of POIs and services that we have in the scenario.		

Requirement ID: REQ_SPM_002	**Category**: Functional	**From use case**: UC_SPM_001 UC_SPM_002
The WASP platform MUST have a (web-based) form to specify simple POI and service profiles.		
Description:		
This is a must because this is the most basic way of specifying profiles.		

Requirement ID: REQ_SPM_003	**Category**: Functional	**From use case**: UC_SPM_001 UC_SPM_002
The WASP platform SHOULD offer a tool that helps the 3rd party service provider to specify more complex profiles.		
Description:		
This is an extension of REQ_SPM_002, its not vital but it would be very nice and maybe even handy.		

Requirement ID: REQ_SPM_004	**Category**: Functional	**From use case**: UC_WBS_001 UC_WBS_002 UC_TOR_001 UC_SPM_003 UC_SPM_004
The platform must be able to store POI and service profiles.		
Description:		
It is necessary to have some entity (dubbed 'registry') that allows for the storage and retrieval of profiles.		

Requirement ID: REQ_SPM_005	**Category**: Functional	**From use case**: UC_SPM_003 UC_SPM_004
The WASP platform MUST allow for users to have different roles.		
Description:		
A service provider has different rights than an end-user, e.g. he can add new or change points of interests in the registry while a regular end-user cannot.		

Requirement ID: REQ_SPM_006	**Category**: Functional	**From use case**: UC_SPM_003 UC_SPM_004
Profiles MUST have associated expiry dates. The registry MUST check for their existence and should enforce them. The registry MAY supply a default expiry date if none was specified.		
Description:		
The scenario specifies that profiles have expiry dates. The registry should at least reject profiles without an expiry date, or insert a default expiry date (for backwards compatibility). For the scenario it is not required that the dates are enforced.		

Requirement ID: REQ_SPM_007	**Category**: Functional	**From use case**: UC_SPM_003 UC_SPM_004
Services MUST have associated schedule information (this service is available weekdays from 8:00 till 16:00), a service MAY be available 24/7. The registry MUST check for their existence, it MAY supply a default schedule if none was specified.		
Description:		
The registry should check that service profiles contain schedule info and reject profiles that do not contain this information, or insert a default schedule (for backwards compatibility). The registry cannot enforce the actual scheduling, this is up to the service itself.		

4.2 Non-functional requirements

Requirement ID: REQ_NF_001	**Category**: Non-Functional	**From**: projectplan
The WASP platform SHALL follow international standards as much as possible for web service description, invocation, and registries as well as for the profiles of users, points of interests and services, and internal standards should be developed such that they are flexible and easily extensible.		
Description:		
• Web service description: WSDL • Web service invocation: XML, SOAP over HTTP • Web service registries: UDDI • Profiles: RDF, RDF-S		

Requirement ID: REQ_NF_002	**Category**: Non-Functional	
The WASP platform SHALL handle all requests in a best-effort manner.		
Description:		
The WASP platform will provide its services in the best way maintainable.		