Utrecht University

Master's Thesis

# Tracing Requirements in an Insurance Software Development Company

*Author:*
Rick Kok

*Supervisors:*
Dr. Fabiano Dalpiaz
Dr. Sergio España

*A thesis submitted in fulfilment of the requirements for the degree of Master of Science*

*in the*

Master in Business Informatics
Department of Information and Computing Sciences

July 2016

# Declaration of Authorship

I, Rick Kok, declare that this thesis titled, 'Tracing Requirements in an Insurance Software Development Company' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

*"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."*

Dave Barry

UTRECHT UNIVERSITY

# *Abstract*

Faculty of Science
Department of Information and Computing Sciences

Master of Science

**Tracing Requirements in an Insurance Software Development Company**

by Rick Kok

In literature, a significant amount of studies has been conducted on the topic of requirements traceability. However, little empirical evidence exists that is convincing from a commercial ROI perspective for adopting traceability. This research compares the as-is situation in an insurance software development company with the revized to-be situation in which more advanced traceability mechanisms have been introduced. By conducting an experiment using data from production environment, the authors showed that these mechanisms lead to both an improvement in efficiency and a decrease in effort in requirements management. Although this is likely to increase the implementation effectiveness in the longer term, traceability is a major challenge for the future.

# *Acknowledgements*

For this thesis, I want to thank Fabiano Dalpiaz, Garm Lucassen and Sergio España *(first supervisor, extra supervisor and second supervisor from Utrecht University)*, as well as Barry Nijkamp *(supervisor at the case study organization)*, for their sharing of time and knowledge for this study.

# Contents

# List of Figures

# List of Tables

# Acronyms

**BPAAS** Business Process As A Service. 2, 15, *Glossary:* Business Process As A Service

**FO** Functional Overview. xviii, 38, *Glossary:* Functional Overview

**PDD** Process Deliverable Diagram. 7, 33, 39, 54, *Glossary:* Process Deliverable Diagram

**RDLC** Requirements Development LifeCycle. *Glossary:* Requirements Development LifeCycle

**RFC** Request for Change. 33, 35, 38, *Glossary:* Request for Change

**RM** Requirements Management. 1, 2, 6, 15–20, 24, 31–33, 39–41, 88, *Glossary:* Requirements Management

**RS** Requirements Specification. 24, 33, *Glossary:* Requirements Specification

**RT** Requirements Traceability. xiii, 1, 4, 8, 13, 14, 17, 30, 91, 92, *Glossary:* Requirements Traceability

**SDLC** Systems Development Life Cycle. 32, *Glossary:* Systems Development Life Cycle

**TO** Technical Overview. 38, *Glossary:* Technical Overview

# List of Terms

**Business Process As A Service** A combination of Software, Platform and Infrastructure as a Service that allows organizations to automate a business process in the cloud [1]. xv, 2, 15

**Functional Overview** The Functional Overview is the name of a document which is the result of requirements management at the case study organization. Main content is a functional decomposition of a requirement, or part of a requirement, that reflects it as complete and precise as possible, with as low as possible ambiguity, in a functional description. xv, xviii, 38

**Process Deliverable Diagram** A notation that combines a metamodel of a process with a metamodel of the data; allowing for a structured overview if both the processes, data and internal relationships are needed [2] [3] [4]. xv, 7, 33, 54

**Request for Change** A means of proposing a change to any component of an IT Infrastructure or any aspect of an IT Service. It may be a document or record in which the nature and details of and the justification and authorization for the proposed change are entered.. xv, 33

**Requirements Development LifeCycle** The life of a requirement, from the moment it becomes known to a company until the moment it is fully implemented.. xv

**Requirements Management** "The activities of gathering, identifying, revising and organizing incoming requirements from the various stakeholders"[5]. xiii, xv, 1, 15, 20, 31, 88

**Requirements Specification** The document in which is defined which requirements will be implemented in which release, and which will not be implemented. xv, 24, 33

**Requirements Traceability** "The ability to describe and follow the life of a requirement, in both a forwards and backwards direction (i.e., from its origins, through its

development and specification, to its subsequent deployment and use, and through all periods of on-going refinement and iteration in any of these phases"[6]. xiii, xv, 1, 13, 91

**Systems Development Life Cycle** A term used in systems engineering, information systems and software engineering to describe a process for planning, creating, testing, and deploying an information system [7]. 32

**Technical Overview** The Technical Overview is the name of a document that contains a technical low-level design of the Functional Overview (FO), created for developers to strictly develop the requirement as designed. xv, 38

# Chapter 1

# Introduction

## 1.1 Motivation

The study on Requirements Management (RM)[1] has been asking some interesting questions for decades, gaining significant knowledge in the field. Traceability has been and still is one of the main challenges; in Figure 1.1, the current state of art in traceability is shown [8]. Requirements Traceability (RT) (referring to the possibility to follow the life of a requirement, both in forward and backward direction, see [6]) as a subdiscipline of requirements management is considered being the next level of RM maturity [9]. As improving a maturity level generally implies investing significant resources into improving processes, it will not be a surprise that it this is true for implementing traceability, too. Therefore, it is a major challenge to satisfactory adopt traceability. But what are the practical gains of implementing better traceability? Performing the literature study, the authors of this study found that little is known regarding the actual gains of adopting traceability. Because of that, the main question that this study asks is: 'What are the actual gains of implementing traceability?'. It tries to show using an empirical test what are the gains in terms of efficiency, effectiveness and perceived difficulty comparing the two situations.

## 1.2 Context of the Research: Insurance Software Development Company

We embed this research in a software development organization, where requirements are managed at a *structured* maturity level [9]. The case study organization is market

[1] Please refer to the Acronyms section for an explanation of the abbreviations

leader in a Business Process As A Service (BPAAS) product for insurers that automates process of business between their clients (insurance companies) and end-users. The organization is growing significantly by performing takeovers in Western- and Northern Europe. To illustrate this, in 2015 year the growth in personnel was 250%. Table 1.1 shows the current (2015) details of the case study organization. Growing at a rate this high poses other challenges; for example in streamlining procedures or managing requirements. According to literature [9], a possible solution to improve RM is to increase traceability. However, since transformations of procedures are often costly, the case study organization wants to know the tangible benefits of implementing traceability. First of all the question is if improving traceability helps solving current RM problems; and to what extent. The case study company has one main software product, that is elaborated in Section 1.2.

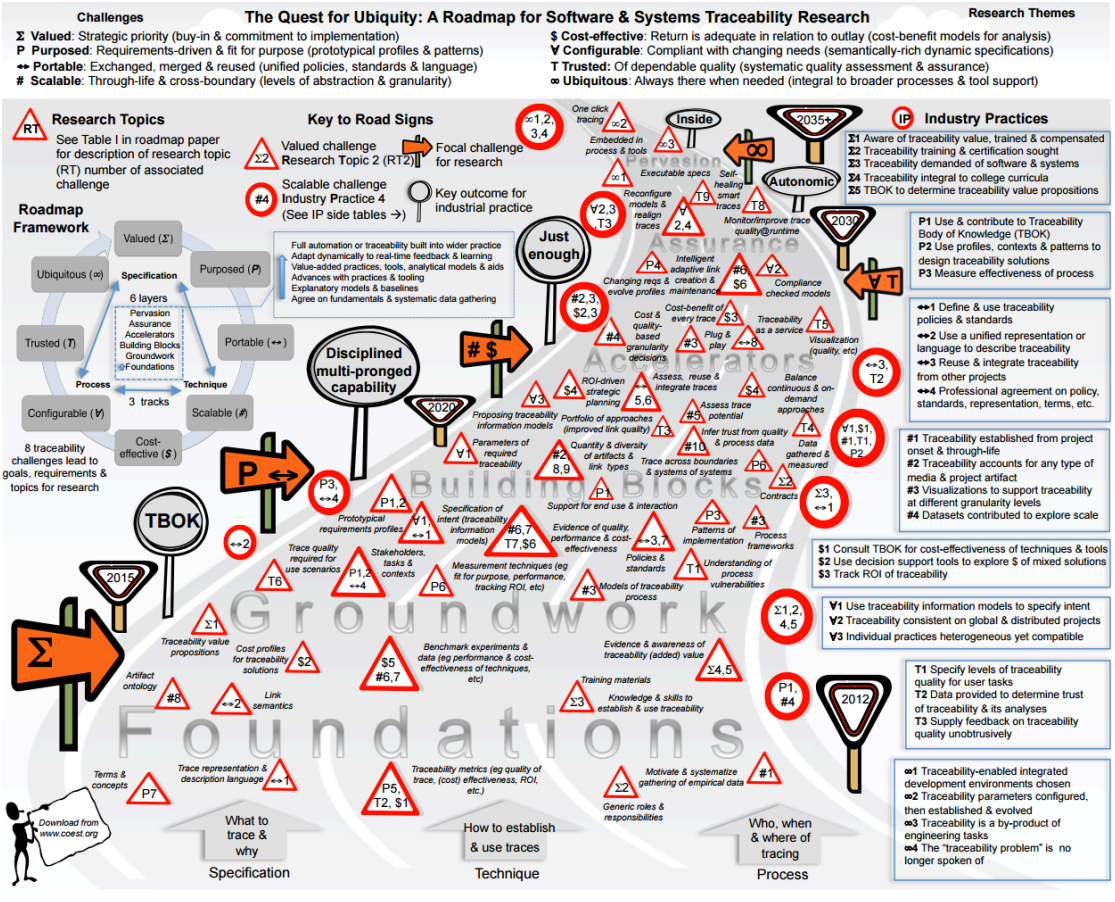



FIGURE 1.1: Roadmap of Current Traceability Research State as concluded by [8] pictures the current state of the field of requirements traceability.

### The Insurance Solution software product

The main product of the case study organization is a solution that automates the process of accepting clients and providing insurance. In the older situation, to get an insurance, one had to submit a form, which is sent to a department within the insurance company. There, some checks would have been performed to decide if the one is able to get the insurance and how much the costs shall be. If the one is accepting the conditions, there has to be a signature, and the insurance starts, giving that one has been honest in filling in the forms. By collecting the business rules for deciding if one is accepted for an insurance and for what price, the majority of cases can be automated. This saves the insurance company a significant amount of time and work. The product supports interaction with other products. For example, using the license plate of a car, it can automatically connect to a government database with information about that car. Other possibility is a check for fraud; if a client is known for a criminal past, he or she will not automatically pass the insurance application. Apart from interacting with customers and databases, the product provides an automation of workflows within the insurance company, by using automatic form handling for example. The focus areas of this product are life insurance, non-life insurance and pension software. This product is created for insurances for all except for life and pension. Examples are: In Figure 1.2, a screenshot

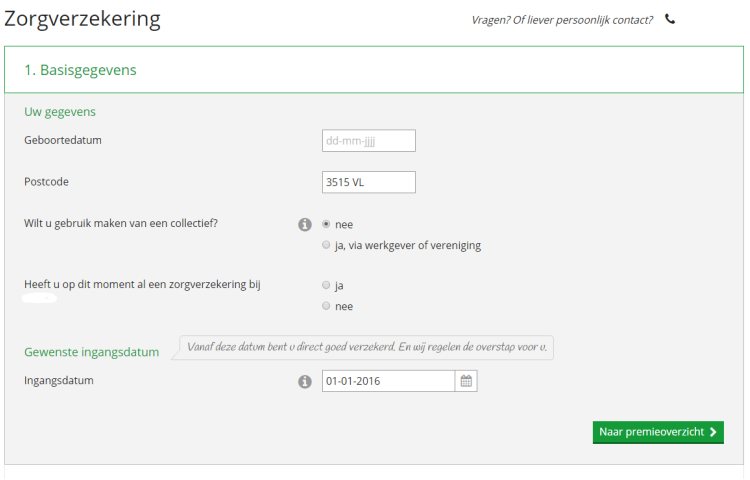



FIGURE 1.2: Screenshot of the Insurance Software Product of the Case Study Company.

| | |
|---|---|
| Name | Keylane |
| Business | Market leader in insurance and pension software |
| Foundation Year | 2000 (as *Quinity BV*) |
| Head Office Location | Utrecht, Netherlands |
| Offices in | NL, BE, DE, SW, NO, GB |
| Number of employees | 420 |
| Turnover 2014 (€) | 46M |

TABLE 1.1: Characteristics of Case Study Organization

is provided as example for an online health insurance form, which is part of the non-life insurance product.

## 1.3 Objectives of the Thesis

The goals (long-term) and objectives (short-term) of this research describe what it is designed to achieve. The business objective is to show IT companies the (possible) advantages of traceability; the business goal is to convince more IT companies to successfully adopt a higher level of RT. If there is more attention to RT from business viewpoint, the business side may be willing to invest more in research. The scientific objective is provide tangible results that may spark more interest in this field; the scientific goal eventually is to allow a framework to exist that enables to predict benefits by level of traceability. To show the reader a short context to the problem, a technique is used from journalism, which is called the *5 W's*.

**Who** The problem affects the case study organization; but since a large number of similarities exist within the software developing domain, in theory it could be generalizable to a number of similar organizations. It should be noted, however, that this study focuses on an agile developmentment method.

**What** The problem is, that requirements implementation is not performing well. This results in an increase in costs and time, an increase in risk, and ultimately, an increase of losing competitive advantages.

**When** The problem occurs during implementation of requirements in the verification phase (within post-RS domain). Although procedures are adhered to, the acceptance test results are not always positive.

**Where** The issue is occurring during development of requirements at the case study organization. It is in the main product, at all locations.

**Why** It is important that the impact of this problem decreases.

The problem statement is:

*Formal Problem Statement:*

WITHIN THE CASE STUDY ORGANIZATION WHICH PERFORMS AGILE INSURANCE SOFTWARE DEVELOPMENT, EFFECTIVENESS IN REQUIREMENTS REALIZATION[a] IS TOO LOW.

[a]Realization refers to the amount of requirements that has been implemented and accepted within time and budget.

The question in this statement is; will improving traceability decrease the number of problems currently experienced at the insurance software case study company? In Section 1.2, the case study company is explained to provide a context for generalizability.

## 1.4 Research Method

Hevner and Chatterjee concluded in [10] that "[...] IT researchers must create innovative IT artifacts that address important problems, demonstrate the capabilities of such artifacts, and evaluate and predict their potential benefits and risks". The method section follows these guidelines. Because this study is based on the principle of observing an experimental situation while influencing one variable, the method of performing a case study has been selected [11]. First, the current method in the case study organization will be analyzed. This is referred to as M0. Next, it is studied what improvements are needed for improving the traceability. After this is applied to M0, M1 is created, which is M0 with improved traceability. The study ends comparing M0 and M1, which will lead to the conclusion if traceability improves efficiency and if it improves effectiveness. From the research objective, in order to find a solution to the problem statement, the main research question has been formulated incorporating traceability as possible solution to the problem statement.

Because this study is based on the principle of observing an experimental situation while influencing one variable, the method of performing a case study has been selected [11].

*Main Research Question*

To what extent does improving requirements traceability also improve the implementation effectiveness in the product software domain?

Two hypothesis are posed. The *null-hypothesis*, 'H0', predicts that no actual correlation is present between the independent and the dependent values. The 'H1' hypothesis predicts that there is a positive correlation between the independent and the dependent variable.

*Hypothesis H0*

Increasing Requirements Traceability does not increase implementation effectiveness.

*Hypothesis H1*

Increasing Requirements Traceability increases implementation effectiveness.

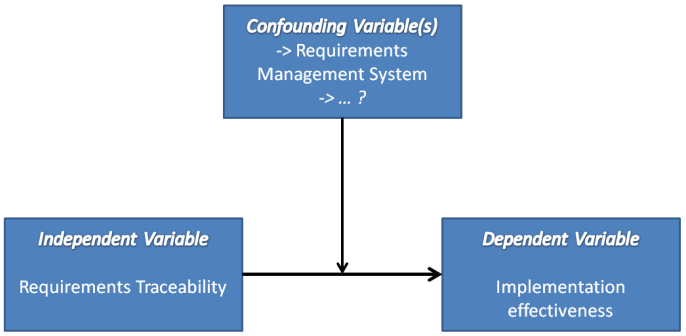



FIGURE 1.3: Research Setup

Figure 1.3 shows the research setup. The independent variable (left; *Requirements Traceability*), is the value that will be manipulated. If the hypothesis is true, the dependent variable (right; *Implementation effectiveness*), shows a positive correlation. There can be influences from *confounding variable*s, also influencing the dependent variable. This can decrease the scientific value of research, because the correlation may be different than concluded. In testing the effect of manipulating the independent variable, it is best to limit the influence of confounding variables. In research however, it is not always possible to know and, thus, include all of these variables [12]. In improving traceability, the directive is to stay as close to M0 as possible. This is done by only improving traceability related aspects in low-level environment; the high-level structure of the method shall not change. The three types of variables can be summarized as follows:

**Independent variable** Traceability of requirements in current RM method [2]

**Dependent variable** Effectiveness of requirements implementation [3]

**Confounding variable** Other aspects that influence the implementation effectiveness which can occur if the experiment is not setup properly. Such a variable can be introduced when changing a method, for example if the testperson has more affinity with a new method and scores better, the researches may be falsely concluding that it was only the independent variable.

The effect of the independent variable on the dependent variable highly influences the conclusion of this study. Table 1.2 shows the type of correlation and the effect on the two hypotheses. If there is a positive correlation between the increased traceability and the effectiveness on requirements implementation, the H0 hypothesis will be rejected and H1 will be true.

[2] This is measured using the number of traceability paths and the quality. See Figure 5.1 on page 59 for more information.

[3] This is measured by the time and effort that it takes to solve a number of tasks that represent the process of implementation of requirements management. Chapter 5 on page 57 elaborates on this topic.

### Sub-questions

For answering the main research question, a total of five of sub-questions is created. For this, the study of [13] is followed, which provides guidelines for improving traceability. For doing this, first of all the structure of the current method, which is referred to as M0, has to be found. After this, the most important problems in requirements management have to be identified, then they will be classified and prioritized. An analysis will tell which problems are likely to be solved by improving traceability. Then the traceability will be improved, and the study ends with an experiment that will try to tell if it can be reasonably shown if improving traceability has a positive effect on the number of encountered problems. Section 1.4 visualizes the guidelines of [13] and, next to these steps, the sub-questions of this research. Since the tool is available and will not be changed, there is no need to follow these steps. Note that in the fifth sub-question, an experiment will find out whether increasing traceability will improve implementation effectiveness, and that the experiment produces results and advices. In a later stadium this can be used for changing SD policies. However, the scope of this study does not allow to implement the changes, and therefore it only partly corresponds to the last phase of requirement traceability adoption.

*First Sub-question.*

WHAT IS THE STRUCTURE OF THE CURRENT RM METHOD (M0)?

**SQ 1.** This sub-question tries to find the structure of M0. It is done using annotation techniques such as a class diagram, followed by the Process Deliverable Diagram (PDD), for making the traceability links visible.

*Second Subquestion.*

WHAT ARE THE MOST SIGNIFICANT PROBLEMS CONCERNING M0?

**SQ 2.** This subquestion will find out what the causes for low implementation effectiveness are in M0. It is done by performing semi-structured interviews; the domain experts are asked what problems are significant with respect to requirements management, and after rating and prioritizing them, a list of main problems will conclude this subquestion.

TABLE 1.2: The Effect of Correlations between Independent and Dependent Variables on the Hypothesis

| Correlation | H0 | H1 |
|---|---|---|
| Positive | False | True |
| Neutral | True | False |
| Negative | False | False |

*Third Subquestion.*

What are the problems from SQ2 that can be mitigated by improving traceability in M0?

**SQ 3.** Using the problems found in SQ 2, the main causes for low implementation effectiveness are identified. This question asks if, and how and why traceability will solve these main problems. The problems that are likely to be solved by increasing traceability, will be used to answer the part of the main research question if traceability can be used for increasing implementation effectiveness.

*Fourth Subquestion.*

In what way can traceability in M0 be improved?

**SQ 4.** This subquestion studies how M0 can be improved from a viewpoint of traceability. Using the knowledge from literature and the vision of domain experts, a traceability-enabled version of M0, M1, is created.

*Fifth Subquestion.*

Which problems from SQ3 does traceability solve and to what extent?

**SQ 5.** This sub-question is responsible for finding out if increasing traceability increases implementation effectiveness, and what other variables (efficiency/perceived difficulty) it may influence. It is done by using M0 and M1 in an experiment, where participants have to solve a number of assignments. The results and other variables are compared, and the conclusion will be drawn if traceability indeed influences other variables.

**Conclusion** The five subquestions will provide an answer to the main research question. In Figure 1.4, the relationship between the main phases of the research (the methods and the experiment at the top of the figure), the chapters (the left axis of the

Table 1.3: Sub-questions compared to RT Adoption Phases by [13]

| Requirements Traceability Adoption Phase | Research Question |
|---|---|
| Recognize RT problems | SQ1&SQ2 |
| Formulate traceability goals | SQ3 |
| Develop methods | SQ4 |
| Acquire tools | *already available* |
| Develop tools | *already available* |
| Change SD policies | SQ5 *(partly)* |

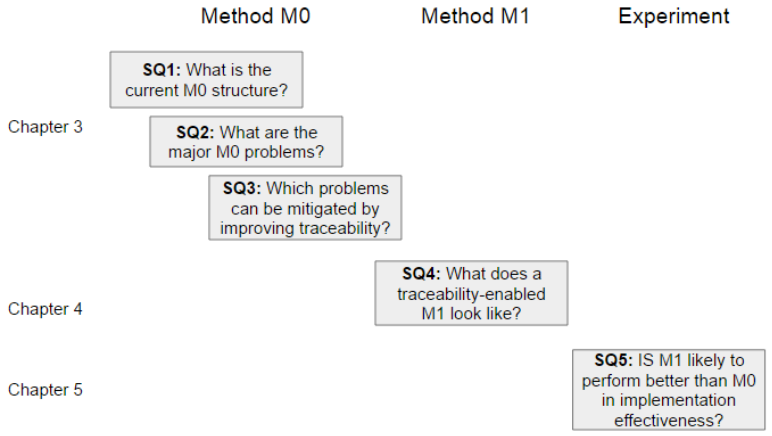



FIGURE 1.4: Research Approach

figure) and the five sub-questions are explained. The first three questions analyze M0, the fourth question has M1 as deliverable and the fifth sub-question concludes the main research question. The horizontal alignment of the subquestions gives an impression what phase of the research is completed. It starts at M0, continues through M1 to the experiment.

## 1.5 Structure of the Thesis

This study is structured in two parts. After the introduction, which is in the current chapter, the analysis part (Part I) starts. In this part, first the literature study is performed (Chapter 2). Chapter 3 analyses the current method, and ends with an overview of the current method, the structure, and the traceability related problems. Then, Part II of the study takes over; which incorporates the design science. In Chapter 4, the M1 method is designed and explained. In Chapter 5, the two methods are compared using an experimental setup. Chapter 6 performs a deeper analysis of the results of the experiment. After that, Chapter 7 lists the threats and validity, before Chapter 8 concludes the thesis. Finally, in Chapter 9, future research directions are outlined.

# Part I

# Analytical Study of the AS-IS Method

# Chapter 2

# Literature Study

The literature study "provides background information needed to understand [the] study (i) assures [...] readers that [the authors] are familiar with the important research that has been carried out in [the authors'] area, (ii) establishes [the authors'] study as one link in a chain of research that is developing and (iii) enlarging knowledge in [the authors'] field" [14]. The structure of this chapter follows these guidelines: after a concise introduction to the field, we explore in detail the relevant aspects of traceability. The technique known as 'snowballing' has been used to perform this literature study [15]. One of the advantages of this technique, is starting at a well-known paper and following the references, resulting in a more efficient literature study [16]. The selected paper to start with is 'An analysis of the traceability problem' by [6], because it has 1237 citations *(at 09-2015)*, which is currently the highest amount of citations in the field. Using the scientific search engine *Google Scholar*[1], it was possible to set a filter from 2011 for the purpose of discovering more recent articles, because of the fast changing nature of this field. The snowballing has been performed mainly in forwards direction, because of the date of publishing of the paper of Gotel and Finkelstein [6].

## 2.1 Definition of Requirements Traceability

A number of definitions of Requirements Traceability (RT) have been proposed in literature.

**Definition 2.1.** "[Requirements traceability] provide[s] a relationship between the requirements, the design, and the final implementation of the system."[17]

[1] `http://www.google.com/scholar`

**Definition 2.2.** "Discover the history of every feature of a system [so that the impacts of changes in requirements can be identified.]" [18]

**Definition 2.3.** "the ability to describe and follow the life of a requirement in both a forwards and backwards direction (i.e., from its origins, through its development and specification, to its subsequent deployment and use, and through periods of on-going refinement and iteration in any of these phases)" [6]

Definition 2.1 focuses on the relationships between conceptual artifacts in the process of requirements management. Definition 2.2 describes RT from a time perspective, it tells about the discovery of history rather than creating links. Definition 2.3 is more precise by taking into account the directions of traceability. This is useful, since traceability is not only backwards (e.g. from low to high level), but can also be forward (e.g. from high to low level). Therefore, in this study, the definition of RT from [6] is used.

## 2.2 Conceptual Model of Requirements Traceability

In [19], a meta-model of traceability is given, which is visualized in Figure 2.1. An object, which represents a requirement or part of it in any state, is documented by a source. The object traces to another object, and a stakeholder has a role in this object. The same stakeholder manages the source which documents the objects.

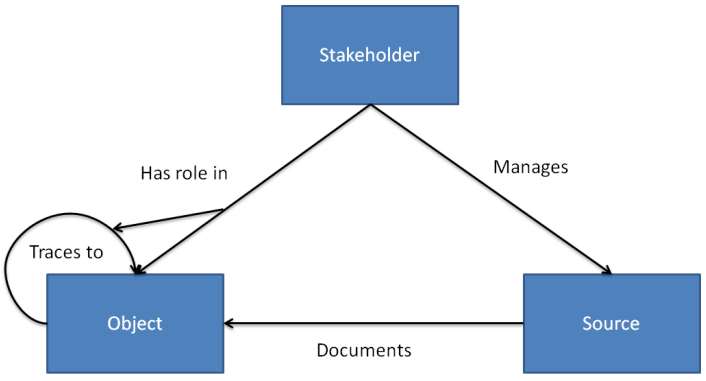



FIGURE 2.1: Meta-model of requirements traceability as identified by [19]. The object refers to an object in requirements management, e.g. a document in the process, which has a source and a stakeholder.

An example of an application for Figure 2.1 is the following. Say, the product manager manages the product, and an additional function has to be added in the form of an API. That API needs a number of changes in the current base system. If compared with the figure, the product manager has the role of stakeholder. He manages the source; in this

case the software product. The object documented by the product is the new API. The stakeholder, which is the product manager, has a role in both the API as he has in the different object that the API is linked to.

## 2.3 Traceability to Cope with Increasing Complexity

Modern software is increasingly complex. To give an illustration; a few of current trends in software are: (i) Artificial purposes: prediction using software, for example it can identify trends [20] such as after an online purchase, an e-commerce application can recommend other products based on your data [21], or (ii) Business Process As A Service (BPAAS) is the technique which allows cloud based software to perform (parts of) business processes [1]. Since business processes can be complicated, for example because of size, work-flows or human cooperation, the software used to replace it has to reflect the exact process as good as possible. In tailor-made software, it is created specifically for the client, thereby reflecting the processes. However, in the case study organization, a standardized BPAAS solution is created that works for all clients, having only one codebase. An example of what the product does Instead of the costly process of sending a letter to the insurance company, which would think about it and calculate to send a letter back, the case study organization was among the first to automate the process. For an insurance company, it means that a significant part of the admission procedure can be performed via the internet. This provides a great competitive advantage for the insurance company.

Since software systems are growing in size and complexity, it is harder for core developers to know and remember the rationale by heart. Therefore, precise Requirements Management (RM) becomes more important; managing requirements is easier when traceability is facilitated [22]. An example of software complexity following the increasing complexity of hardware is displayed in Section 2.3. It plots the mean amount of transistors in personal computers against the lines of code in the Debian OS[2], which is displayed by the dashed line. Although these are just examples, it is safe to assume that software systems will increase in complexity as processing power does.

The need for improving RM nowadays is rapidly growing [23, 24]. It is recognized that a more effective RM requires good traceability practices [25], while some even argue that it is a separate phase in RM maturity [9]. Furthermore, according to a study of [26], agile development in general needs traceability.

[2] `https://www.debian.org/`

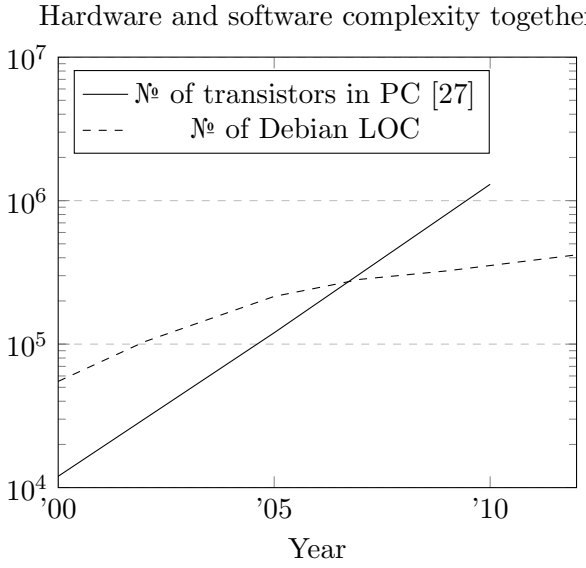



FIGURE 2.2: This figure illustrates the raising complexity in the field of Information Science in the near past, to provide the reader with a tangible idea about the current state of technology.

The topic will be on measuring the change in effectiveness of requirements handling using a RM method without focus on traceability and an RM method with focus on traceability.

As the complexity of software systems is rising and organizations are transforming into higher IT-dependency, the pressure for developing within a predefined time span and budget becomes higher. A large number of studies and practical experiences show that this is not easy. According to [28], requirements management is one of the most important factors that influence development time and costs within requirement. The impact of repairing a wrongly implemented requirement is increasing significantly during later phases of development. It has also been concluded that (i) requirements are likely to be the most common class of error and (ii) requirements are likely to be the most expensive errors to fix, therefore the solution lies within good requirements management.

## 2.4 Advantages of Traceability

In a software development project, traceability is vital for its ability to store and link rationale to requirements. RM can be facilitated by capturing information necessary to understand requirements evolution and verification [29]. It is named to help ensuring other software qualities, such as adequacy and understandability. If traceability is neglected, it can lead to less maintainable software and to defects due to inconsistencies

or omissions [30]. This is supported by [31], where it is stated that the ability to allow changes to any artifacts – requirements, specification, implementation– to be traced throughout the system is an important property of any systems description technique. Traceability is helpful, especially if it can link designs to justifications, important decisions, assumptions from the past and the contexts in which design solutions are made [32]. In another study, RT is considered to be a quality attribute of a system. It is a characteristic a system should possess and include as a non-functional requirement [33]. Some authors view traceability as the ability to "discover the history of every feature of a system" so that the impacts of changes in requirements can be identified [18]. In [34], the authors analysed literature and found out that in the topic of RT, interest is increasing. It is said that because of the often far-reaching impacts of requirements, it is hard to find all system components that are affected by a requirements change [35]. Davis [36] supports this by concluding that the later in the development cycle a software error is detected, the more expensive it will be to repair. It is easier to assess the impact of a proposed change if a roadmap is available that shows where each requirement is implemented. Moreover, traceability is the only means to ensure that the source code is consistent with its requirements and that all and only the specified requirements have been implemented [37]. In [38], the authors state that traceability links are typically created and a maintained using a RM tool, in a word processor or spreadsheet programme. Traceability helps in keeping track of (i) parentage, (ii) interconnections and (iii) dependencies among different requirements [35]. A study of [39] compared 17 methods of implementing traceability. For comparing these methods, a number of parameters are defined. Here a selection of the parameters is listed that is relevant to this study:

- Coverage
  The coverage refers to the area where traceability is implemented. The coverage is either:

  Origin $\rightarrow$ Requirement,

  Requirement $\rightarrow$ Requirement,

  Requirement $\rightarrow$ Other Artifact or

  Other Artifact $\rightarrow$ Other Artifact.

- Tool Support
  Tool support refers to the tools that can be used for supporting the method.

- Level of Detail
  This refers to the types of requirements that are traceable. The requirement is either

Functional requirements

Refers to the requirements that affect functions of a system. For example; it has to support external API's.

Non-functional/Quality requirements

This type refers to the attributes of a system. For example security, uptime, usability.

### 2.4.1 Typical Need for Traceability

Most software producing organizations started small. Where small projects are suffering less in terms of an absence of good requirements management, costs will rise exponentially in larger software products. When the the number of requirements increases, a structured RM approach might not longer be sufficient. This manifests itself an unpleasant result[3]:

> **Customers are not satisfied with the end result of the implementation of a requirement, although all phases of the implementation have been agreed upon.**

According to [6], the source of the problem can be either (i) external requirements elicitation or (ii) internal requirements traceability. The first is responsible for the wrongly interpreted requirements before specification phase. This is the phase in which the requirements only exist in the customers' heads. The second, internal requirements traceability, refers to the traceability from the specification towards the end-product.

According to [40], Eliciting often proves to be a difficult task, common problems being:

- problems of scope, in which the requirements may address too little or too much information;
- problems of understanding, within groups as well as between groups such as users and developers; and
- problems of volatility, i.e., the changing nature of requirements.

### 2.4.2 Traceability Facilitates Finding Orphan Code

Mentioned in [35], orphan code is code that is of no use anymore. Without traceability, it would be very hard to find out. With traceability, all the fragments of code that are

[3]This knowledge was gathered interviewing number of internal experts at the case study company

deprecated can be found and removed. In [41], unclear traceability was studied yielding the next conclusions:

(i) The reconstruction of the design rationale through analysis might be expensive.

(ii) Design criteria and environmental factors that influence the architecture might be unclear.

(iii) Business goals and constraints might be ignored.

(iv) Design integrity might be violated when intricately related assumptions and constraints are omitted.

(v) Tradeoffs in decisions might be misunderstood or omitted.

(vi) The impact of the changing requirements and environmental factors on a system could not be accurately assessed.

### 2.4.3 Traceability Reduces the Cost of Repair

Leffingwell and Widrig (described in Table 2.2) summarized the results of a study in which three large organizations individually measured the costs of repair of a fault in software development [28]. It has been tested during the phases of requirements, design, coding, unit test, acceptance test and maintenance. It is remarkable that the three tests yielded roughly the same results. The left column lists the concerned stages, the right lists the relative costs of repair. The stage of *coding* is the reference value. This is supported by a number of similar studies [23, 28, 36, 42]. When a change is made into existing code, it is much easier to find dependencies [35]. Research into RM practices has been performed by IAG consulting[4] [42]. It yielded a number of interesting results. Among the key findings, some are valuable to this study:

(i) Companies with poor business analysis capability will have three times as many project failures as successes.

(ii) 68% of companies are more likely to have a marginal project or outright failure than a success due to the way they approach business analysis. In fact, 50% of this group's projects were "runaways" which had any 2 of:

Taking over 180% of target time to deliver

Consuming in excess of 160% of estimated budget

Delivering under 70% of the target required functionality

[4] `http://www.iag.biz`

(iii) Companies pay a premium of as much as 60% on time and budget when they use poor requirements practices on their projects.

(iv) Over 40% of the IT development budget for software, staff and external professional services will be consumed by poor requirements at the average company using average analysts versus the optimal organization.

(v) The vast majority of projects surveyed did not utilize sufficient business analysis skill to consistently bring projects in on time and budget. The level of competency required is higher than that employed within projects for 70% of the companies surveyed.

When read literally, a requirement can be seen as a required need. The definition of [43] is used for defining the meaning of a requirement:

(i) "a condition or capability needed by a user to solve a problem or achieve an objective

(ii) a condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard,specification, or other formally imposed document

(iii) a documented representation of a condition or capability as in (i) or (ii)"

This survey shows the importance of a good requirements management method. The survey was performed selecting 400 random IT-projects, after which 110 where manually selected according to the demands. One of these demands was the size of the project; it had to be more than $250,000 in size to ensure a level of complexity. It should be noted that 'Business Analysis' is comparable with a super-requirement; or the other way around: a requirement is a low-level business analysis. It has been discovered, that when not taking the analysis seriously, the failures versus successes ratio will be three to one. This can hardly be a sustainable practice. If the business analysis is performed better, but not very good, project will not fail completely, but rather it will be more costly in terms of time and money. If focusing on RM, the findings show that poor RM will consume over 40 percent of the IT development budget; which could have been avoided.

TABLE 2.1: Five Requirements Management Maturity levels as identified by [9].

| Maturity Level | Description |
|---|---|
| 1 | Written |
| 2 | Organized |
| 3 | Structured |
| 4 | Traced |
| 5 | Integrated |

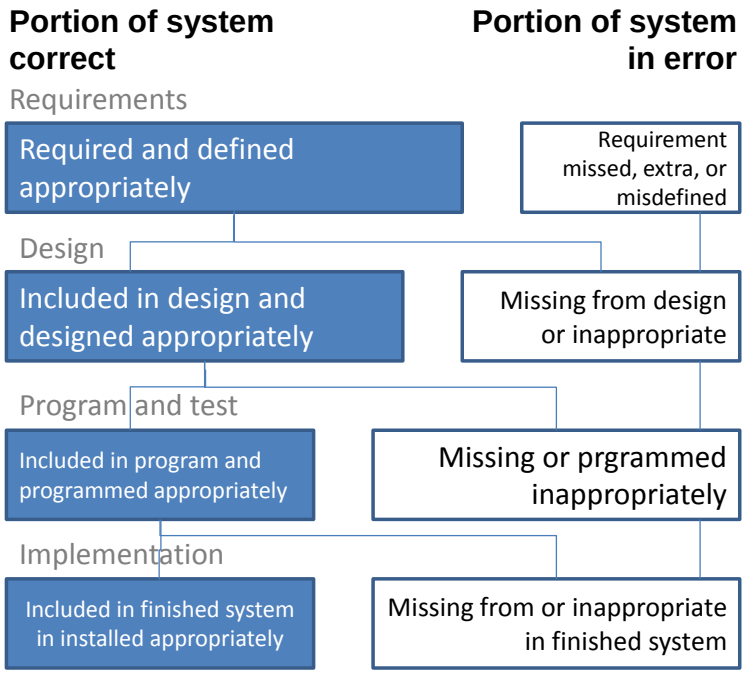



FIGURE 2.3: Impact of Requirements Management Errors Increasing during Process (adapted from [23]).

Research of Goldsmith [23] shows the effect of missing, extra or misdefined requirements throughout the project. Figure 2.3 on page 21 shows this effect: the blue and white rectangles respectively represent the parts of the IT-project which are performed as wished, and the parts that can be considered a failure due to the consequences of the missing, extra or misdefined requirement. As the error impact starts in the *Requirements* phase; it increases in volume at the expense of the 'correct' rectangles. This means, that when using the wrong requirements, larger parts of the system will be affected in later phases.

## 2.5 Phases and Maturity of Traceability

This section explains what traceability is, and provides the reader with some important related information to understand the importance.

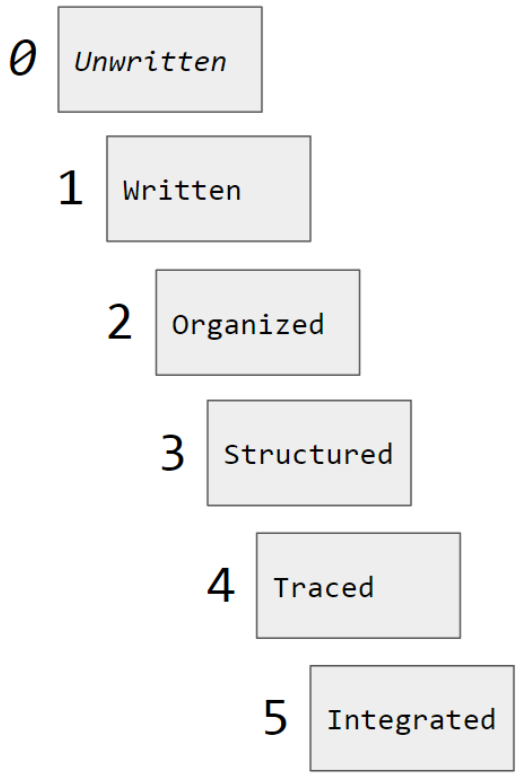



FIGURE 2.4: Five Levels of Maturity in Requirements Management [9].

### 2.5.1 Maturity in Requirements Management

Multiple authors recognize the need for a CMM for RM [9, 24, 44]. The maturity model which is described in this section can be traced back to an older, widely accepted Capability Maturity Model from the Software Engineering Institute [24]. They state that this maturity model is used in 101 countries, by 11 national governments and it is translated in 10 languages. As a significant part of the world is reportedly using it, the value of the concept is proven. In [45] however, the value is doubted and a combination of the CMMI with IBM's RMM [9] is proposed instead of the maturity model of the SEI. The five maturity levels are summarized in Figure 2.4.

**Level 0: Unwritten** The very first level is described by using the number zero, as, according to the authors, it means no maturity exists at all. It is identified by a failing product combined with inability to provide value to the customer.

**Level 1: Written** Here, requirements are stored in a safe environment (e.g., with backup functionality). Advantages are:

(i) Real basis for contract with a customer

(ii) Enabling developing team to work more effective, allowing work to be divided among team members

(iii) As a source for requirements is available, new members can more easily understand the system

**Level 2: Organized** In this level, requirements are *organized*. This encompasses a central storage which is accessible by all users, possibly secured. Moreover, requirements should be properly formatted, allowing for testing and understanding by all stakeholders. Requirements documents should be more or less standardized, rendering this level important to both the content and the markup of requirements. Version control ensures that delivered work will be more accurate to the requirements specifying it by preventing outdated requirements from being built. Advantages of adopting this level are "less rework and better customer acceptance".

**Level 3: Structured** This level is about *structuring* the requirements. This helps in understanding them better, by categorizing in functional/non-functional, features/software requirements and customer/market/user requirements. A non-functional requirement (also referred to as *Quality Attribute*), influences the basis of the system more than a functional requirement, for example when the system has to update a database system-wide within a second; this might be of large influence to the operational core of the system. Another important aspect of this third level is prioritization; this creates a larger perceived customer value by delivering the most important requirements in earlier stages.

**Level 4: Traced** For this study, it is the most interesting level; it is about *traceability*. The authors state that "In a systems engineering environment, this hierarchy starts with system requirements and moves down to subsystem requirements, program requirements, and element requirements. [...] The ability to keep track of these relationships is commonly called traceability, and it entails identifying and documenting the derivation path (upward) and allocation/flow-down path (downward) of requirements in the hierarchy." [9]. This creates a number of advantages:

(i) Understanding how one requirement influences others

(ii) Determining if requirements are complete coverage analysis

(iii) Sophisticated reporting functionality.

**Level 5: Integrated** Due to the scope of this study, it will only be elaborated shortly. Integrated RM means that requirements are incorporated in the complete software development process. This process ensures that all requirements will be implemented, moreover, no requirements is allowed to change without review and

approval. Implementation of requirements or changes are tested against the requirements. Project leaders should have access to the status of the realization of the requirements in a project.

### 2.5.2 Pre- and Post Specification

Gotel and Finkelstein studied the RM process in [6] and found a requirements specification to be an important separation between de pre- and post Requirements Specification (RS) phase. According to the authors, it is hard to provide an all-encompassing solution to the challenges that RM offers. A significant number of problems in RM and –traceability were analysed and a number of solutions proposed. This result was adapted by España in [46] to include traceablity links. Figure 2.5 shows the result. In the middle, the *Requirements Specification* separates the phase of elicitation, which can be seen left of the RS (Pre-RS traceability), and the phase of developing the requirement into a system, which can be seen on the right (Post-RS traceability). The first, *Pre-RS*, refers to the phase of requirements management which consists of retrieving the knowledge that a customer has in mind, documenting it, and verifying that knowledge. When the customer agrees, the requirement will be specified in the RS. The requirements specification refers to a document that contains the requirements for a system, general and functional. It is a document that summarizes for both parties (the client and the software company) what they can and should expect. When the RS is approved, the requirements are implemented into the code. Usually, a functional and technical design are among the artifacts created, as are functional documentation, technical documentation, code, and possibly more. The latter, *post-RS*, creates traceability among these artifacts which are either created or adapted because of the requirement. The extend to which these changes and creations are traceable to their origin and the other way around are referred to as traceability.

Different techniques can be used. Davis performed an empirical study into the effectiveness of these results, and found that a (structured) interview is the most effective method [47].

## 2.6 Best Practices from Literature

The literature study found that a number of studies are interesting for the experiment that is performed in Chapter 5. This section lists a number of best practices that are vital to this paper.

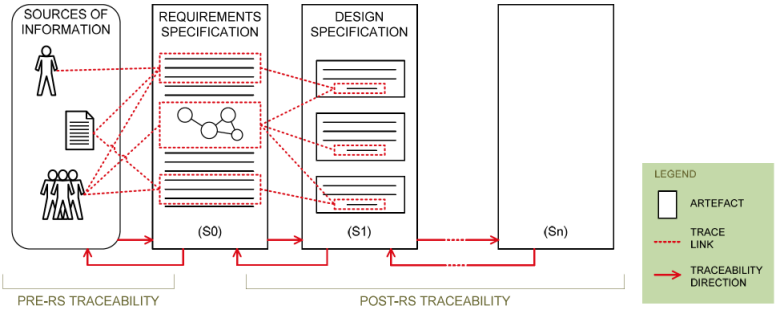



FIGURE 2.5: Internal *(Pre-RS)* and External *(Post-RS)* Requirements Management, in the center the Requirements Specification *(RS)* [46].

### 2.6.1 Depth of Traces

It is generally hard to maintain traceability [47]; among others the seemingly high costs for organizations is a reason for neglecting or abandoning traceability efforts. In [48], it is stated that any traceability effort should start with the question: what is the main aim/purpose of the traceability data? In [49], it is recommended that if trace data is not needed in order to meet a specific goal, it should not be collected and/or stored .

### 2.6.2 Traceability within Requirements Management Maturity Indices

In [9], requirements management maturity was studied and five levels were found, showed in Table 2.1. A higher level means more mature requirements management, which has a number of advantages in effectiveness. In [24], different Requirements Management Maturity Indexes (RMMI's) were compared, and it was concluded that a common RMMI that failed to mention traceability in fact contained it. The topic of requirements traceability was studied in [6], and it was concluded that there is a difference between two phases in requirements engineering. It was stated that there is both a pre-requirements specification (pre-RS) and a post-requirements specification (post-RS) phase. This is also known as the 'validation' and the 'verification' phase [50]. Validation is about ensuring that the requirement in the RS reflects the original, as the customer wants it ('do we build the right system'), the latter means the extent to which the requirement is built as it should according to the RS ('do we build the system right'). Due to the scope, this thesis only focuses on the 'verification' which is in the 'post-RS' phase.

Leffingwell and Widrig in [28] end their first chapter concluding that requirement errors are:

(i) likely to be the most common class of error, and

(ii) likely to be the most expensive errors to fix.

The authors continue telling that the frequency of errors and the multiplicative effect make it easy to predict that requirements errors will contribute to the majority, which can be 70 percent or more, of the rework costs. Since the rework typically consumes 30%–50%, total costs of requirements errors can easily consume 25%-40% of the total budget.

### 2.6.3 Impeding Factors to Requirements Traceability

In [6], Gotel and Finkelstein listed common problems with requirements traceability extracted from various sources. Among problems imposed by providers are:

- Perceived as optional extra.
- Little benefit result for large amount of work.
- All stakeholders need to adopt traceability, individual efforts are ad-hoc and localised.
- Absence of management for maintaining traceability; roles are to obtain and document the required information; organise it; and maintain it.
- If the culture is deliverable-driven, the quality of traceability information can be low.
- If the information is structured, being traceable in many ways, it does not guarantee being up to date.

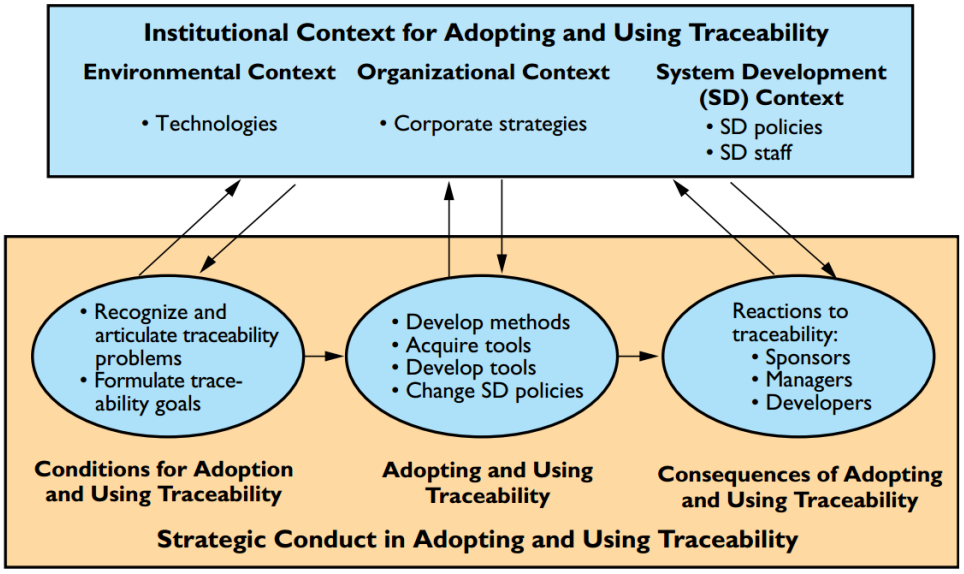
The end users, on the other side, also encounter problems. Among them are:

- It is unclear which quantity, heterogeneity, and depth of detail of the potential information is required.

Table 2.2: Relative Cost of Repair per Development Phase in Software Development [28]

| Stage | Relative Cost of Repair |
|---|---|
| Requirements | 0.1-0.2 |
| Design | 0.5 |
| Coding | 1 |
| Unit test | 2 |
| Acceptance test | 5 |
| Maintenance | 20 |

FIGURE 2.6: Factors Affecting Requirements Traceability Practice [13]




- Often there is reliance on personal contact, stakeholders use that when something that is out of date, undocumented, inaccessible, or unusable.

In [13], the practical influences into requirements traceability have been studied. The effects of external and internal influence has been monitored for over four years, in order to "develop reference models to guide improved practice, and understanding the factors that facilitate or impede traceability practice". This is especially useful for this study, by providing guidelines to implement a new requirements management method. Figure 2.6 shows the results of this study. The authors defined three different situational contexts from which influences to requirements traceability practices are found. These are: (i) Environmental context (ii) Organizational context and (iii) System development context.

**Environmental Context** This context contains the means of technology that allow for requirements traceability. An example is the use of different CASE *(Computer Aided Systems Engineering)* -tools that can be tightly integrated. It should be noted that the authors differ between the high- and low-end users. Steps for achieving traceability from this context are (i) Recognize and articulate traceability problems; especially for the users that don not perceive a great number of benefits it is important that problems are recognized, and (ii) Formulate traceability goals. This context is referred to as *Conditions for Adoption and Using Traceability.*

**Organizational Context** This context is about corporate strategies. One should be careful for the low-end users; if there is no or little perceived benefits from the

extra work that arrives when actively promoting traceability; "lack of organizational commitment to a comprehensive traceability practice is a common trait to this group". High-end users however, are said to understand that traceability provides more space for achieving competitive advantage. Steps for achieving traceability from this context-view are: (i) Develop methods (ii) Acquire tools (iii) Develop tools and (iv) Change system development policies. This context is referred to as *Adopting and Using Traceability.*

**System Development (SD) Context** This third context encompasses the last phase of requirements traceability implementation. Policies have to be changed, staff shall be instructed how to work with traceability. Reactions to traceability are defined as originating from: (i) Sponsors (ii) Managers and (iii) Developers. All the stakeholders should be satisfied with the new traceability-enabled working practice. This context is referred to as *Consequences of Adoption and Using Traceability.*

## 2.7 Preliminary Research Framework

In [13], the authors identified both impeding and facilitating characteristics on the effectiveness of implementing requirements traceability, which are shown by Table 2.3 on page 29. Left, the column *Category* lists the contexts as depicted in Figure 2.6, combined with the *Conditions for Adoption and Using Traceability.* The next column lists the (sub-) concepts that are within these categories, following a one-to-many type of relationship. This also happens in the next column. Last two columns present the facilitating and impeding characteristics. These two are most interesting for this study. Based on this characteristics table, the difference between the current and ideal situation regarding requirements traceability can be identified. In order to fulfill the objective of this study, a number of questions have been posed here for investigation. This framework will be used to address subquestion 2.

- Source-view (Physical artifact where traceability information is maintained)
- Stakeholder-view (Management level studies usually focus on the stakeholder perspective, which is about agents involved in the management of traceability)
- Object-view (Object-oriented view of a requirement in which the requirement is an object)

TABLE 2.3: Impeding and Facilitating Characteristics for Requirements Traceability *(Adapted from [13])*

| Category | Concept | | Facilitating Characteristics | Impeding Characteristics |
|---|---|---|---|---|
| Environmental context | Technologies | | Tailor or develop new technologies | Inability to effectively use available technologies |
| Organizational context | Corporate strategy | | Organizational commitment for quality system development | Traceability as a mandate |
| System development context | System development policies | | Standardized methodologies | Ad-hoc practices |
| | System development staff | | Project staff | External staff |
| Conditions for adoption and use of traceability | Traceability problem | | Mechanisms for process improvement | Required overhead |
| | Traceability goals | | Capture process/product dependencies | Sponsor/standards compliance |
| Adopting and using traceability | Develop methods | | Well defined, across all phases, links to stakeholders | Ad-hoc, select activities |
| | Develop/acquire tools | | Tailored, embedded in system development environment | Stand-alone, incompatibility among tools |
| | Change system development policies | Costs | Life cycle view of costs | Treated as an overhead |
| | | Scope | Selective capture | Uniform capture |
| | | Implementation strategy | Incremental adoption | No clear strategy |
| | | | Adequate training and support | Inadequate incentives and support |
| | | Human resource policies | Tangible and intangible benefits | Inadequate incentives and protection |
| | | | Use of traceability for quality assurance | Use of traceability for performance appraisal |

## 2.8 Conclusion

In this chapter, a summary from literature is provided that helped the authors gaining background information in the field of RT. For the reader, it placed traceability in a broader context. With this context in mind, Chapter 3 continues with analysing the current requirements management method, M0.

# Chapter 3

# Analysis of the Current Method (M0)

This chapter describes how the current method (from here referred to as M0) is analysed. It starts with a positioning between other Requirements Management (RM) methods in Section 3.1. Next, the activities are studied in Section 3.2. The next step is analyzing the relationships between the processes and concepts in M0, which is done in Section 3.3. In Section 3.4, the problems of M0 are analyzed using a number of expert interviews. The main problems will be categorized in Section 3.5. Conclusion of this chapter can be found in Section 3.6

## 3.1 Positioning M0 between Other RM Methods

The RM method which is used in M0, show similarities to a number of common cycles from literature.

### 3.1.1 V-Model

The M0-method, being the current RM method, is not significantly different when compared to other known methods. One example of this, is the V-model, first proposed by Rook [51]. Figure 3.1 shows this model. When M0 is compared to Figure 3.1, the overlap is easy visible, as seen in Table 3.1. Although the phases in different models do not contain the exact same sub activities, the order of phases is the same. For generalizability, this is important since a large number of RM methods are similar to the V-model.

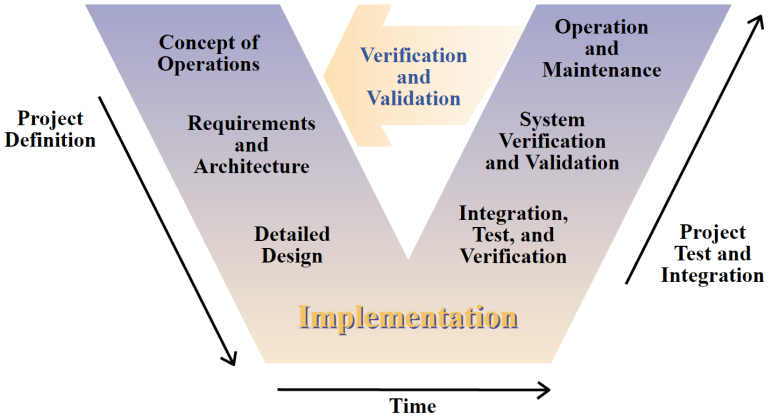



FIGURE 3.1: The V-model [51]

### 3.1.2 Systems Development Life Cycle

When M0 is compared to the general A term used in systems engineering, information systems and software engineering to describe a process for planning, creating, testing, and deploying an information system [7] (Systems Development Life Cycle), it is more or less similar. However, mainly in the first part of Figure 3.2, some differences can be identified. In M0, the first phase of the requirements management trajectory is a global analysis (the M0 structure can be found in Figure 3.4 on 36). In Figure 3.2, the first activity is 'Planning'. The opinion of the author on this matter, is that the original Systems Development Life Cycle (SDLC) is more waterfall-oriented, allowing the 'planning' phase to be first [52]. In M0, planning and analysis are more intertwined.

## 3.2 Activities

The current structure of the RM method is identified using observations and interviews. The result is shown by Figure 3.3. The representation used is a class diagram, because

TABLE 3.1: Overlap of M0 method and the V-model [51]

| Step | M0 phase | V-model phase |
|---|---|---|
| 1 | Global analysis | Concept of Operations; Requirements and Architecture |
| 2 | Functional analysis and design | Detailed analysis |
| 3 | Technical analysis and design | Detailed analysis |
| 4 | Development | Integration, test and verification; |
| 5 | Test | System verification and implementation |
| 6 | Implementation | System verification and implementation; Operation and Maintenance |

of the higher flexibility and the global function of the representation. The figure is explained as follows: First, the requirement is analyzed by a number of experienced stakeholders. They decide what impact the requirement will have in terms of time for realizing. If less than three hours, it is an enhancement (which does not need to walk the traject of design documents et cetera), otherwise it will be handled and built as a regular requirement. Note that since in Section 2.5.2 the Requirements Specification (RS) has been used as seperation between the pre- and post RM phase, the phase referred to as *1. Out of scope* is explicitly is out of scope for this study. The next activities are performed in the Request for Change (RFC) phase, where the designs are created and validated. Next, the *5. Code* activity incorporates the changes in code, which enables the new functionality in 'QIS', the main software product. At the same time, documentation is updated, which is referred to as *2. Documentation*. After this analysis, the official documentation was analysed, for the purpose of a more precise analysis including different artifacts. Figure 3.3 is used as input for creating a Process Deliverable Diagram (PDD).

## 3.3 Analysis of Relation between Processes and Concepts

For better understanding the current RM method, we continued analyzing M0 using the official documentation for the method. Table A.1 on page 94 shows this procedure. It is detailed, and contains all necessary steps for requirements management and implementation. However, when this official procedure was checked against the knowledge of internal domain-experts, it was told that these procedures are more used as guidelines

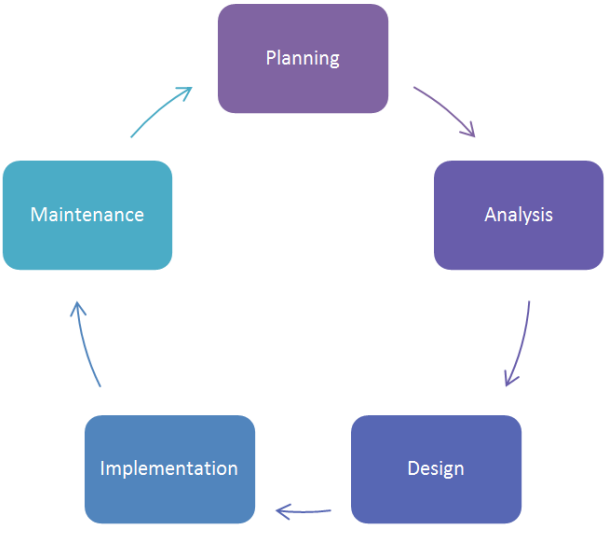



FIGURE 3.2: Systems Development Life Cycle

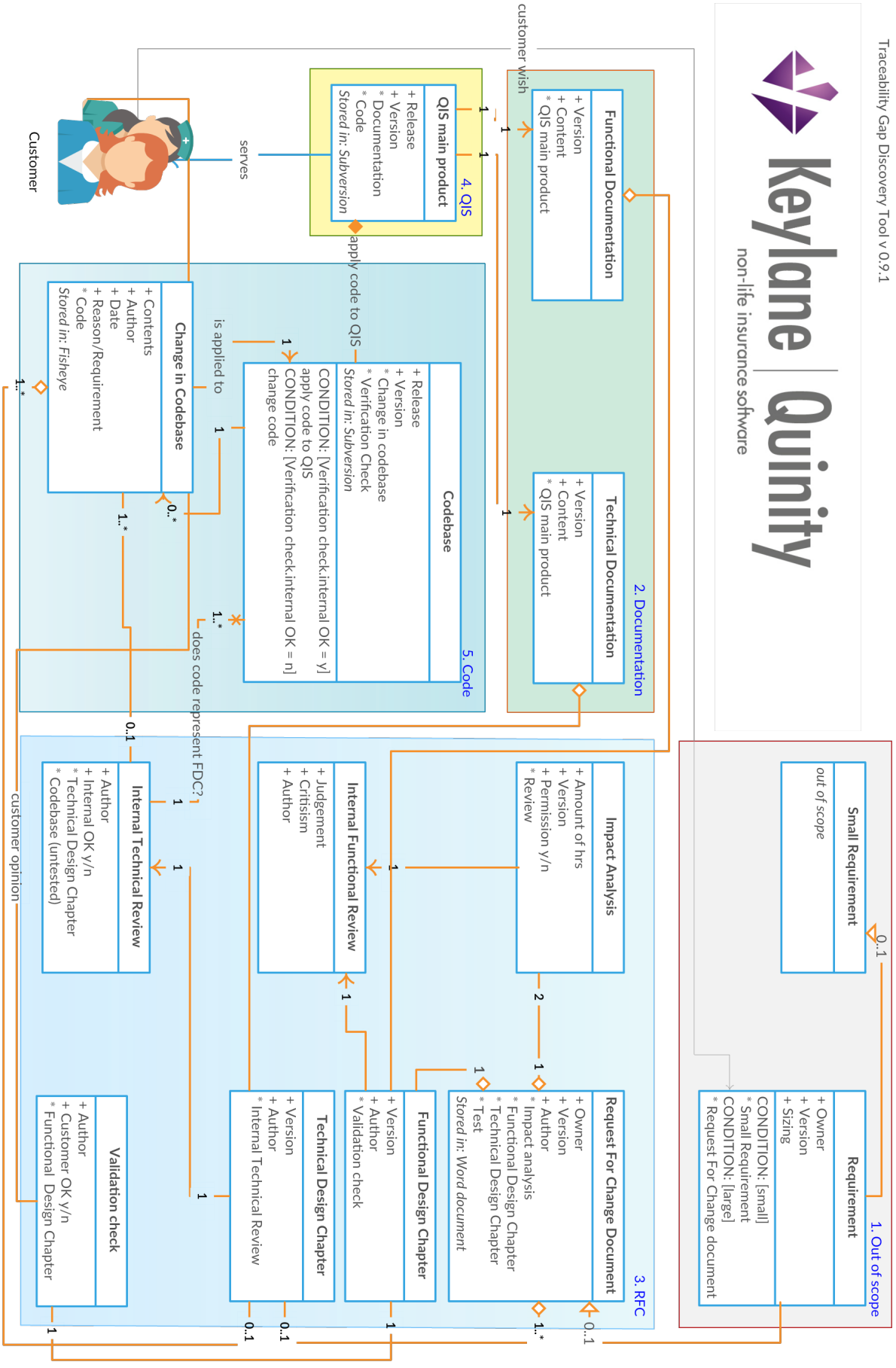



FIGURE 3.3: Class Diagram displaying Changes because of Requirements in M0.

than strict rules. For this reason, the authors created an overview of processes and deliverables, as shown in Figure 3.3 on page 34, updated with the knowledge of experts in this domain. Because it is vital for the traceability in this study to use an overview that best represents the method used by stakeholders, which may not fully adhere to official standards, Figure 3.4 on page 36 is used for this study. This allows us to bring current processes closer to the prescribed/official method.

### 3.3.1 Analysis of Processes

Six major phases have been identified in M0 which can be seen in Figure 3.4. The activities have been described in Activity Table Table 3.2 on page 38. To provide the reader with more understanding regarding the process, The next paragraphs are describing the different phases from top to bottom. When a requirement is feasible and the impact is approved by the client, it is built according to the procedure that is described in Figure 3.4.

**1. Global Analysis & Planning** In this first phase, the requirements are analysed. It will be studied, to decide what solution is preferred. Is it a client-only solution? Maybe a bug is not a bug but a feature? And if there is a decision to fix a bug, how should it be fixed? The analysis is done carefully, but in the least possible amount of time. When the stakeholders agree on the value of a feasible solution, the client is presented the sizing of this solution. If the client agrees, an RFC is created that, in later phases, contains the traceability and links to data and design decisions for implementing this requirement.

**2. Functional Analysis & Design** If the client agrees with the sizing, it means that the organization can bill the client for realizing the requirement. This starts with creating a functional breakdown of the requirement. Generally it takes a few talks between stakeholders to discuss how to best realize the requirements. After consensus is reached, the requirement is split up in a number of small requirements. For all of the small requirements there will be a functional design. When completely designed, an experienced team-member will be asked to review the functional overview. Then the functional analysis is to be signed by the client; this is done to find out if the requirement has been understood correctly. Afterwards, the client has to approve the functional design to verify that the functional designer understood the clients' wishes.

**3. Technical Analysis & Design** If the client agrees with the functional design, the development moves on to the phase of technical analysis and design. Per functional

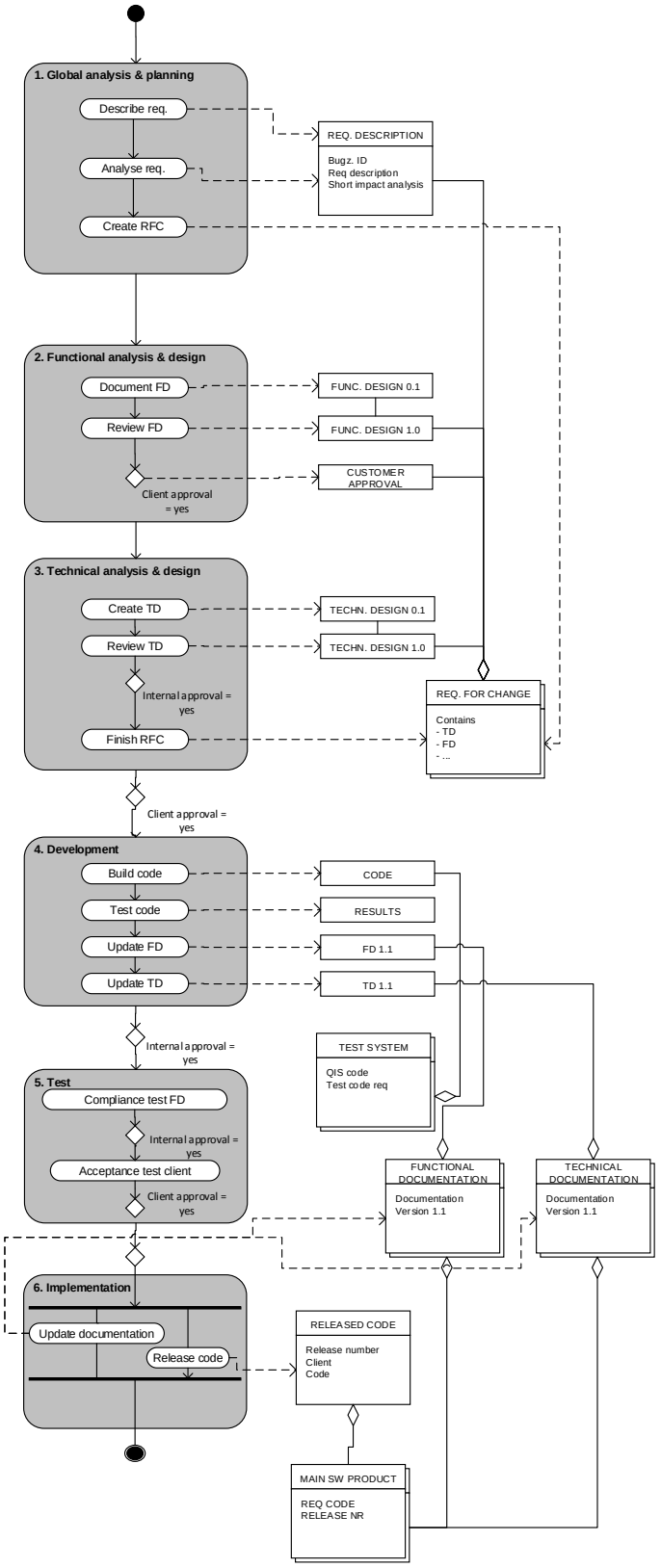



FIGURE 3.4: A PDD involving both Processes and Deliverables from the M0 Method.

sub-requirement, a technical design is made. This boils down to programming without actual programming; it is design how to program it, without developing. Afterwards, an experienced team-member reviews the results to spot abnormalities and faults.

**4. Development** This phase involves the software engineers, they are building the code for implementing the re1quirement according to the technical design.

**5. Test** Generally, there are four different tests. First of all, the developer checks if the code is functional and works without errors. After that, it is tested whether the developed code reflects the technical design; later on it is checked against the functional design, and finally, there is an acceptance test for the customer.

**6. Implementation** If the tests have been passed, the next step is to implement the new functions into a release. There are four major releases per year, it is generally patched into the first next release. Because of the character of the software, which means avoiding tailored client-specific additions as much as possible, several times a year all client specific additions are patched into the main product.

### 3.3.2 Analysis of Concepts

The concepts, displayed in Figure 3.4 are explained by Table 3.5. In this table, the left column shows the concepts from the figure, which are described in the right column. Because the details of low-level documents are not of high value in this stage, there will not be an extra elaboration. What is of high value, is the relationships between the different documents.

The authors examined the relationships between the different concepts carefully, and found out that the most important difference for traceability is the cardinality between the documents and a central node. Table 3.3 on page 39 shows the identified cardinalities. Using this information, a figure is created for clarifying the structure. Figure 3.5 on page 40 shows the system artifacts, the central change node and the change artifacts. The cardinality between the document and the central node is defining for the type. For example; the system documents are identified by the the $1 \rightarrow n$ relationship with the central node, where process documents are identified by the $n \rightarrow 1$ cardinality. System documents describe the system and how the different requirements are implemented in the system (hence the $1 \rightarrow n$ cardinality with the requirement). Process documents, on the other hand, are the documents that are produces for realizing the requirement. For

this reason, the cardinality is $1 \rightarrow n$ between the requirements and the describing process documents. The central node in this table is the issue-tracker, in this case 'Bugzilla'. It contains rationale, decisions, traceability links, links to stakeholders et cetera. Since most of the traceability links can be retrieved using this node, it plays an important role in preserving traceability. Table 3.4 shows the current situation between documents. In view of forwards traceability, it generally classifies as partly. Partly means that some of the documents may be referenced to, but not per se. In practice, the authors found that it highly depended on the accuracy of the person that created the different documents. From the central traceability artifact to the global analysis, however, was almost in all

TABLE 3.2: Activity Table belonging to Figure 3.4

| № | Activity | Sub-Activity | Description |
|---|---|---|---|
| 1. | Global analysis & planning | Describe requirements | A short decription involving the stakeholder and requirement context is put in the issue tracker. |
| | | Analyse requirements | This phase involves a global analysis to provide the client with a rough analysis for impact and sizing. |
| | | Create RFC document | This document will contain the analyses from phase 2. and 3. |
| 2. | Functional analysis and design | Document Functional Overview (FO) | First functional breakdown of the requirement is created. |
| | | Review FO | A review is performed by an experienced co-worker for improving the original FO. |
| 3. | Technical analysis and design | Create Technical Overview (TO) | The functional breakdown is explained on low-level technical view. |
| | | Review TO | A review is performed by an experienced co-worker for improving the original TO. |
| 4. | Development | Developing TO | The TO is transformed into code in this phase. |
| | | Test code | Testers test the code for bugs, and if necessary, give feedback for improvement. |
| | | Update general FO | The general (system-wide) FO is adapted to the new change caused by implementing the new requirement |
| | | Update TO | The general (system-wide) TO is adapted to the new change caused by implementing the new requirement |
| 5. | Test | Test system against FO | Testers check if the new version complies with the FO |
| | | Functionality check | The new requirement is checked against the original requirement to confirm that it reflects the agreed upon functionality. Client has to provide approval. |
| 6. | Implementation | | The requirement is added to a new release and is fully usable. |

occurrences fine. The implementation-related documents however, scored the worst on traceability. This can be explained by the fact that in that stage, most of the process has already been executed, meaning that there is a low will to enable traceability in this phase. In Chapter 4, this table is used to show what traceability elements are improved.

### 3.3.3 Preliminary Problem Based on Documentation

In [35], it is said that every requirement should be able to be identified using a unique number, preferably on fine-grained level. In Using BugZilla as central change artifact, this already is the case. However, frequently, traceability links in this node are incomplete or absent. It happens that valuable information is not documented (or referred to) at all.

TABLE 3.3: Cardinality of Traces in Method M0

| *Traceability relation* | № of requirements per artifact | № of artifacts per requirement |
|---|---|---|
| Central change artifact | 1 | 1 |
| Change artifact | 1 | n |
| System artifact | n | 1 |

TABLE 3.4: Current Traceability Situation in M0 per Identified Phase (see Figure 3.4)

| Phase | Traceability to Central Traceability Artifact |
|---|---|
| Global analysis | Full |
| Functional analysis and design | Partly forwards |
| Technical analysis and design | Partly forwards |
| Development | Partly forwards |
| Test | Partly forwards |
| Implementation | Barely any |

TABLE 3.5: Concept Description of PDD of Figure 3.4

| Concept | Description |
|---|---|
| REQUIREMENT DESCRIPTION | Context, stakeholder and client description including first sizing |
| FUNCTIONAL DESIGN | Functional breakdown of requirement |
| CUSTOMER OK | Approval of customer for continuing the RM process |
| TECHNICAL DESIGN | Technical breakdown of FUNCTIONAL DESIGN |
| REQ. FOR CHANGE | Request For Change document that contains all analyses of a requirement |
| CODE | The end product that, when released, provides the new requirement to the client |
| RESULTS | Document containing results of tests |
| TEST SYSTEM | A clone of the production environment for testing purposes |
| FUNCTIONAL DOCUMENTATION | The system-wide describing document for all functional aspects |
| TECHNICAL DOCUMENTATION | The system-wide describing document for all technical aspects |
| CODE | The code that contains the requirement, to be released into MAIN SW PRODUCT |
| MAIN SW PRODUCT | The main product of the insurance software development company |

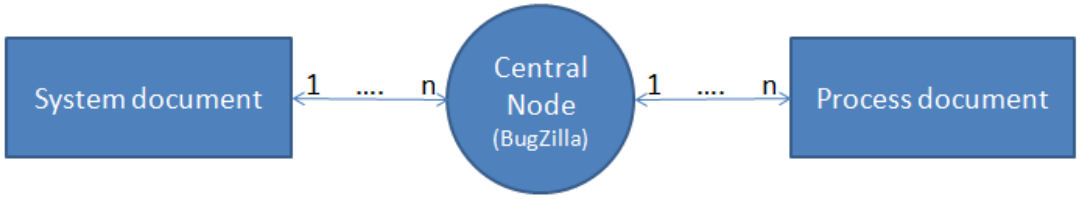



FIGURE 3.5: Meta-model of Traceability Links

## 3.4 Analysis of M0 Traceability Problems

For measuring the effect of improving traceability, first current state of problems in RM has to be defined. Because the domain experts (employees) know best where in the process problems or issues appear, the preferred method is a semi-structured interview [53].

### 3.4.1 Analysis Method

Because surveys can produce invalid results due to the respondents rate, a semi structured interview is the preferred manner for questioning subjects [54]. In [55], two main advantages are named: first, the suitability for exploration of the perceptions and opinions of respondents, even in complex and/or sensitive issues. Answers will be more clarified. Next, because the subjects are picked, the group of subjects will be more varied. The interview was prepared in consultation with the case study organization. It starts with an introduction about the roots and the targets of this study, telling about anonymity. Next, both *requirements* and *requirements engineering* are defined, and at the end of the first part it is explained how to book the hours spent on this study.

### 3.4.2 Interview Setup

In this section, the reader is explained how the semi-structured interview was created, as shown in subsection "The Interview". In the first part, the subject communicates the most common problems to the interviewer. The next part asks for naming the experienced problems including some of the properties, e.g. what, why, impact, occurrence and if the interviewee tells a possible solution, it is included. Because of the structure of the answers form of the semi-structured interview, it does not matter whether the interviewee first lists the problems and elaborates them afterwards, or another manner. Next to that, it adds clarity to make sure that the answers are written down clearly and unambiguously. When the problems were identified and written down, the next part of

the interview started. Using the class diagram (Figure 3.3 on page 34), the interviewee was given time to understand the process on paper.

#### Questions of our Interviews

(i) In which team are you working?

*Choice* Product team Client team

(ii) What is your function in the team?

*Choice* Team leader Func. designer Software eng. Tester

(iii) What is your relation with requirements management?

*Open question*

(iv) Did you ever experience problems regarding the realization of requirements?

*Open question*

When the respondents answered question *iv* positive, the interviewer told them to name the most serious problems they encountered. Using a predefined notation, consistency is guaranteed.

(v) Problem *1 . . . n*

(a) What is the problem?

(b) Why is the problem there?

(c) Impact 1 2 3 4 5 6 7 Occurrence 1 2 3 4 5 6 7

(d) Solution *(If the interviewee tells the solution during the interview)*

### 3.4.3 Stakeholder Types

This paragraph describes the involved stakeholders in the case study organization RM process. This are the stakeholders:

**Team leader** All teams have a team leader. He is responsible for the proper functioning of his team. He is also responsible for the result of the team; in other words, he makes the key decisions in projects.

**Functional Designer** The functional designer (FD) has the task of breaking down the requirement as received from the customer to very precise details. The reason

TABLE 3.6: Interviewees

| | Client Team | MainPr Team |
|---|---|---|
| Team Leader | 2 | 2 |
| Func. Designer | 1 | 3 |
| Software Engineer | 1 | 1 |
| Tester | 0 | 1 |

behind this is that requirements can be large and unclear; when the customer expected an different result, it will be costly to fix.

**Software Engineer** The software engineer (SE) breaks down the functional design into technical design. Combining his knowledge of the systems with the to-be-implemented functional design, the SE creates a technical design. Next, the code is built having the technical design as guideline.

**Tester** The testers perform two types of tests. First of all, it is tested whether the system has been built according to the technical design, for example, it can include a bug which has to be removed. Next, there will be a test against the functional design to see whether the built software actually represents the customers' wishes. If that test is passed too, the software will get permission for release.

**Client *(or* Customer*)*** The client is considered to be the most important stakeholder; without clients there is no business at all.

**Teams** Two types of teams are identified. The client teams work full-time for a client. In The client-specific functionality will be incorporated into the next base release to avoid maintaining numerous tailored additions. In Table 3.6 and Table 3.7, these teams are identified as 'Client'. The second type is the main product team. Their responsibility is maintaining the main releases and incorporating tailored code into it. These teams are identified in the tables as 'MainrPr'. For performing the interviews, first a list was made containing the names of the senior employees, having some experience in their particular fields. From the four team roles and the two teams (see Table 3.6), at least one senior employee was found for interviewing.

### 3.4.4 Results of Interviews

When summarized, Table 3.8 shows the results. The most problems exist in and around the phase of 'Functional Analysis & Design'. This analysis can be concluded with the statement that most M0 problems manifests in the functional analysis and design phase. Section 3.4.5 elaborates on this statement. Finally, some of the interviewees listed

TABLE 3.7: Problem per Function per Interview

| Subject № | Team | Function | Problem № | Category |
|---|---|---|---|---|
| 1 | Client | SE | 1 | 4 |
| | Client | SE | 2 | 4 |
| | Client | SE | 3 | 4 |
| 2 | Client | FD | 4 | 1 |
| | Client | FD | 5 | 1 |
| | Client | FD | 6 | 1 |
| 3 | MainPr | TL | 7 | 3 |
| | MainPr | TL | 8 | 1 |
| 4 | Client | TL | 9 | 3 |
| | Client | TL | 10 | 2 |
| | Client | TL | 11 | 5 |
| 5 | MainPr | TL | 12 | 3 |
| | MainPr | TL | 13 | 5 |
| 6 | MainPr | FD | 14 | 1 |
| | MainPr | FD | 15 | 3 |
| 7 | MainPr | FD | 16 | 2 |
| | MainPr | FD | 17 | 5 |
| 8 | MainPr | SE | 18 | 2 |
| | MainPr | SE | 19 | 1 |
| | MainPr | SE | 20 | 5 |
| 9 | MainPr | FD | 21 | 3 |
| | MainPr | FD | 22 | 1 |
| | MainPr | FD | 23 | 1 |
| 10 | MainPr | Test | 24 | 2 |
| | MainPr | Test | 25 | 2 |
| 11 | Client | TL | 26 | 5 |
| | Client | TL | 27 | 2 |
| | Client | TL | 28 | 2 |
| | Client | TL | 29 | 2 |

*Legend:*
Category 1: Out of scope
2: Functional Design incorrect
3: Functional Design Approval incorrect
4: Documentation hard to find
5: Other reasons

traceability as a possible solution to the problems. This is supported by [23], they state that traceability mainly improves completeness by improving the areas of consistency and dependency.

**10, 18, 24, 25, 28** If traceability between the requirement and the functional design(s) could be improved, it would be easier to check if the FD is satisfying.

**7, 9, 12, 15, 21** In between the client's initial ideas and the FD, some data is lost and corrupted. Better traceability *could* improve this.

### 3.4.5 Additional Findings from the Interviews

This subsection will provide the reader with the remarks that interviewers provided during the interviews, grouped per phase. The number of problem is printed bold, followed by the remark itself. If a problem has been stated multiple times, or if problems are very similar, they are combined to one problem for readability.

#### Category 1: Pre-Requirements Specification Requirements Elicitation *(Out of Scope)*

**4, 5, 6, 8, 14, 19, 22, 23, 27** The scope of this study explicitly is not about what is referred to as *external traceability*, or *pre-RS traceability* [6]. Because of that, this group of problems will not be studied.

**10** "The FD is not clear enough, interpreted wrongly by software engineers." This problem tells that the functional design allows for wrong interpretation. This is because software engineers can interpret the design different, resulting in software that works according to the FD, but does not represent the wish of the clients, being as representative. This is dangerous to the function of the FD, because when the procedure is followed, problems may still exist. The solution, according to the interviewee, lies in recording

TABLE 3.8: Results of First Research Subquestion *(see Table 3.7 for full list)*

| № | Problem Area | № of refs |
|---|---|---|
| 2 & 3 | Functional Design phase | 13 |
| 1 | Out of scope | 8 |
| 4 | Documentation-related | 3 |
| 5 | Other problems | 5 |

the meetings with clients and writing down the conclusions. This encourages the client to verify if the FD is correct.

**18** "The functional design is incomplete." This is a result from either the client which provided incomplete information or the functional designer overlooking details. No solution was provided.

**24** "The requirement is not built according to the customer's wishes." This is due to the functional design not being exhaustive. Different interpretations are possible. No solution was provided.

**25** "The wish from a customer is not fully realized in the FD." This represents a problem, because of the leading function of the FD. The given reason states that the functional designer thinks that the software engineer will understand what has to be done, while reality suggests otherwise. No solution was provided.

**28** "The technical designer doesn't understand the functional designer." This can again be related to the FD; which is the product of the functional designer. This is because the FD is either too simple or too complicated. No solution has been provided.

**Category 3: Incorrect Functional Design Approval**

**7** "When a solution is built according to the FD, the client still disagrees." This problem differs to the ones named in subsection 'Incorrect Functional Design' because here it is assumed that the client approves a functional design that does not reflect his needs. This is either because of a small amount of knowledge in this field, the client possibly does not know how to check the FD. Another reason states that at the client, the managers that approve FD's are not the end-users. This can lead to an unwanted situation of groundless approvals. A possible solution could be prototyping; this provides a simpler method for understanding a FD.

**9, 12, 15** "The FD approval is not signed involving end-users." This interviewee points out that the client's decision-makers sometimes don't understand what the end-users need. This results in a significant more costly project, combined with frustration from both sides. As possible solution, more involvement from an end-user is needed in the requirements analysis traject.

**21** "Functional design is not clear to the client, maybe he didn't even read it." Because it does seem unlikely that a client does not read the document describing what is being paid for, it could be due to other factors such as the inability to understand what is written. The solution could be involving the client more in the requirements management process.

## 3.5 Categorization of the Problems

This section lists the categorization of the problems as identified in Table 3.7, as well as the number of identified problems within this category. Per paragraph, a number of solutions are added that were named by the interviewees. When supported by literature, a reference is added.

**Category 1: Pre-Requirements Specification Requirements Elicitation *(Out of Scope) (8)*** The results contained a number of problems that arise at an early stage in requirements management, or even before the requirements management starts. A problem that has been named five times is the small knowledge clients have regarding requirements and the insurance software system. Both teamleaders and functional designers said: *"Clients don't know what they need!"*. Four identified problems had another focus: *"Always analyse a requirement the customer poses, he is not the party for creating solutions, we are."*. During the interviews it became clear that requirements are approved by the customer and built according to functional design, and when finished, only seem to contain a (very) inefficient solution, if a solution at all, to the original problem. From the knowledge that the client has, however, the solution seemed logical. A solution to this problem, according to the interviewees, is to be stricter in always analyzing the clients wish. Moreover, it was said that the client should be more educated and more involved; which will increase clarity. Another interviewee pointed out that sometimes, a requirement is squeezed and changed into the current system, which in the end may be having more impact than the separate realization of it.

**Category 2: Incorrect Functional Design Creation *(8)*** The next step is the creation of the functional design. One of the interviewees mentioned, that *"if the requirement is divided into multiple FDs, it should be more clear and the collection of functions should be tested well against the original requirement"*. Regarding the FDs, it has been said that: *"they are not exhaustive"*, *"incomplete"* and *"not clear"*. This

results in problems, for example *"the technical designer does not understand the functional designer"*, and *"functional designers and technical designers barely communicate"*, which results in other unwanted situations.

Solutions: On the problem of dividing a requirement into more FDs, the solution is better traceability according to the interviewee. On the problem that the FD is not good enough, traceability has also been named as solution, along with more communication, better reviews and better training.

**Category 3: Incorrect Functional Design Approval *(5)*** A heavy impact problem is the false FD approval by a client. When the client signs the contract for building the FD, they "don't read", "understand", "are not interested" and according to the interviewees it happens that "wrong people have the power to decide". This leads to the situation in which a client pays for software that doesn't fit the real needs; rendering both the software developer and the client unsatisfied.

Solution: The solution to this problem lies in improving multiple areas. First, it should never be assumed that clients have substantial enough of knowledge of the field of IT and thus are able to pose their own requirement. Next, the client should be made very clear during the requirements analysis sessions (in presentations, meetings) what the consequences can be resulting from false approvals. This may include techniques such as involving end-users in the requirements analysis, or by means of presenting prototypes to the customer. This is confirmed in a study by [56].

**Category 4: Documentation *(4)*** Tthe interviewees twice told that "the information in the release notes is unclear for clients", one adding that "it is hard to find the right information".

Solutions: The main solution provided by the interviewee that posed these problems was to improve the traceability in documentation.

## 3.6 Conclusion

Four main problems have been identified within requirements management in the case study organization, which are shown in Table 3.8. From these four, two are within the scope of this study. One phase has the highest amount of problems: the Functional Analysis and Design phase. Some interviewers stated, that improving traceability is likely to reduce these problems, because most are caused by low accessibility of vital information.

# Part II

# Design Science: a Traceability-enabled Method

# Chapter 4

# Designing a Revised Method

According to [48], in designing a traceability solution, the first objective is to find out the main purpose/aim of the data. In our case, the main purpose of the data is to be able to retrieve knowledge; traceability can help in better retrieving knowledge. However, as stated by Cleland-Huang et al. in [49], if trace data is not needed in order to meet a specific goal, then it should not be collected and/or stored. In Chapter 3, we discovered that a number of important documents is not traceable at all; the traceability goal for M1 is to make all changes in all artifacts traceable to the originating requirement. Doing so, we follow the guidelines of [48, 49].

## 4.1 Concepts to Improve

In Section 3.3.2, on page 39, Table 3.4 showed the M0 traceability situation. From this situation, the traceability will be improved to full traceability in all phases. This is, because a request for change may not to start with a global analysis, but for example starts with a technical document that was made for another requirement. Traceability allows the stakeholders to find all related data; eliminating the need for having to search for data. By using the issue tracker 'BugZilla' as central change artifact (see Figure 3.5 on page 40) for interconnecting all data artifacts, for every requirement all artifacts shall be traceable. This is done by two-way traceability; not only it is possible to navigate from the central change artifact to the functional design, also to navigate from a testcase document to the central change artifact (backwards traceability; see Section 2.1). The differences are shown in Table 4.1 on page 52.

The author encountered a number of different levels of traceability. There were cases of no traceability at all, which is classified as 'none'. If the traceability was partial, for

TABLE 4.1: Comparing M0 and M1 Traceability Situations per Phase

| Phase | M0 traceability | M1 traceability |
|---|---|---|
| Global analysis | Full | Full |
| Functional analysis and design | Partially forwards | Full |
| Technical analysis and design | Partially forwards | Full |
| Development | Partially forwards | Full |
| Test | Partially forwards | Full |
| Implementation | None | Full |

example there is a statement of the existence of a document without pointing to the location, it is classified as partially. We also discovered some cases of full traceability in M0. Note that there is forwards and backwards traceability, which has been described in Section 2.1. The current level of traceability per phase is shown in Table 4.1. If a wish for elaboration on the different phases is present, please read Chapter 3.

## 4.2 In-depth Example

To provide the reader with more understanding in the traceability changes, one example reflecting real situations are provided for the three different classifications of traceability, as showed in Table 4.1. Note that for this section, it is assumed that Section 3.3.2 on page 37 has been read. As stated there, the issue tracker is used as central traceability artifact. This means, a traceability link has to be created to— and from all process documents and system documents. Those two types of documents differ, because of the cardinality towards the central traceability artifact (see Figure 3.5 on page 40).

### 4.2.1 System Document and Central Change Artifact

The system documents describe the system, meaning that a high number of requirements are linked to this document. Because both forwards and backwards traceability will be fully implemented in M1, it has to be easy to follow the link from requirement to (exact part within) the system documents and back. Since the system documentation currently is maintained in a word processor[1], it means that the changes in this document have to be accompanied by a comment that links back to the origin of the requirement in the issue tracker. An example of an application is shown in Figure 4.1. The two advantages of using the comment functionality, is that using the built-in search function, on requirement number, all changes that occured to the documentation because of that requirement are visible, while per sentence in the system document, it is possible to see

[1] Microsoft Word

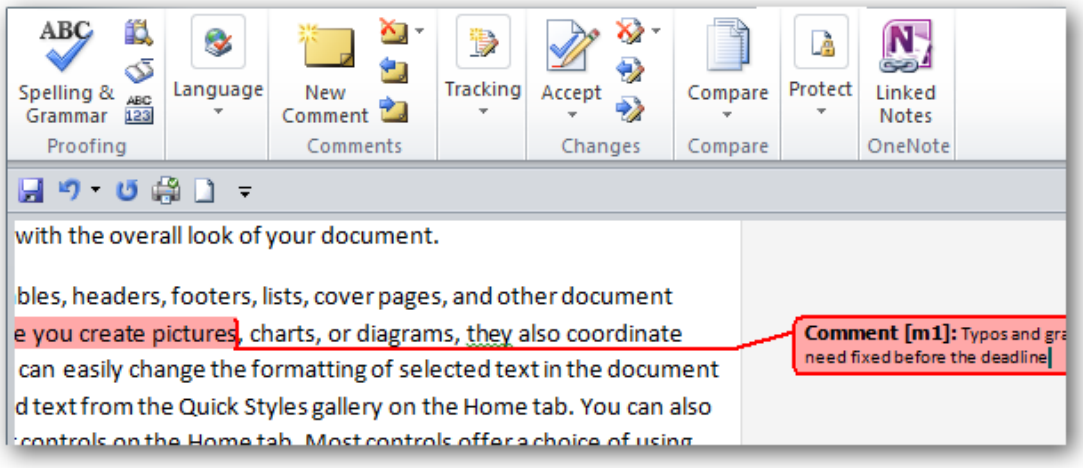



FIGURE 4.1: Screenshot of the Comment Functionality of a Word Processor

which requirement originally led to the change. In M1, a standard structure for linking to a requirement is used. The best notation depends on the important traceability variables, and these may be different in other organizations. In M1, we chose to add the next variables to a change:

- Link to original issue-tracker website
- Number of bug
- Date that the change was performed
- Stakeholder that performed the change.

This information was added in the next notation:

```
LINK: [link]
BUGNUMBER: [bugnumber]
DATE: [date]
STAKEHOLDER: [stakeholder].
```

In the central change artifact, little has to be changed. This is because the comments that are added contain links to the requirements. For this reason, it is possible to search on requirement number and find the reference. For full forwards traceability thus, only a reference to the file location and file type is needed.

### 4.2.2 Central Change Artifact and Process Document

This type of traceability link is one to many, or $1 \rightarrow n$. For each requirement, multiple process documents exist. Among others, these are the functional design, technical design and test report. Since the location of the documents is heavily dependent on the client and the function, it is important to note. However, since one document is for one requirement, a traceability link to the location of the document is sufficient for traceability. Furthermore, it is important to know the type and comments if needed. (The name of the stakeholder is automatically added if he submits a post to the issue tracker, and therefore, not necessarily needed in the traceability link.) The file path and file name are separated, because a hard link to a file ignores updates. When separated, the stakeholder knows which version was originally posted, but can choose to read a newer version, if preferred for a reason. What is chosen for usage in M1, is:

```
FILE: [filetype]
===========
Path: [filepath]
Name: [filename]
([extra information])
```

*Please note that [extra information] can contain data such as such as chapter, date, page, author, versioning et cetera.*

In this manner, documents are easily traceable from the issue tracker. Backwards traceability is easier; since all documents have an introductory chapter that sums up a number of variables such as date, creator and owner, it is easy to add a link to the original issue-tracker page. In the word processor, there is a possibility for creating links, and using this, links have been created that point back to the original central traceability artifact.

## 4.3 Process Deliverable Overview of M1

In Figure 3.4 on page 36, a Process Deliverable Diagram (PDD) of M0 has been created. This figure has been adapted to show the changes made in M1, the result can be seen in Figure 4.2 on page 56. The changes are colored, to show the differences from process and artifact perspective. In green, the added processes and the deliverables are depicted. The green dashed lines reflect the M1 relationships between the processes and the concepts

that are added. The red dashed lines show the M1 traces that are added between the deliverables by the improved traceability.

## 4.4 Conclusion

Improving method M0 to M1 may seem to be a small change to some. Figure 4.2 clearly shows the changes. However, if a small change is all that is needed, it may prove to be a highly efficient change. We carefully analyzed where traceability is missing, studied plans to improve it, and found an easy method of improving it. In Chapter 5, the experiment is explained, and it will tell if the small changes actually have an effect or not.

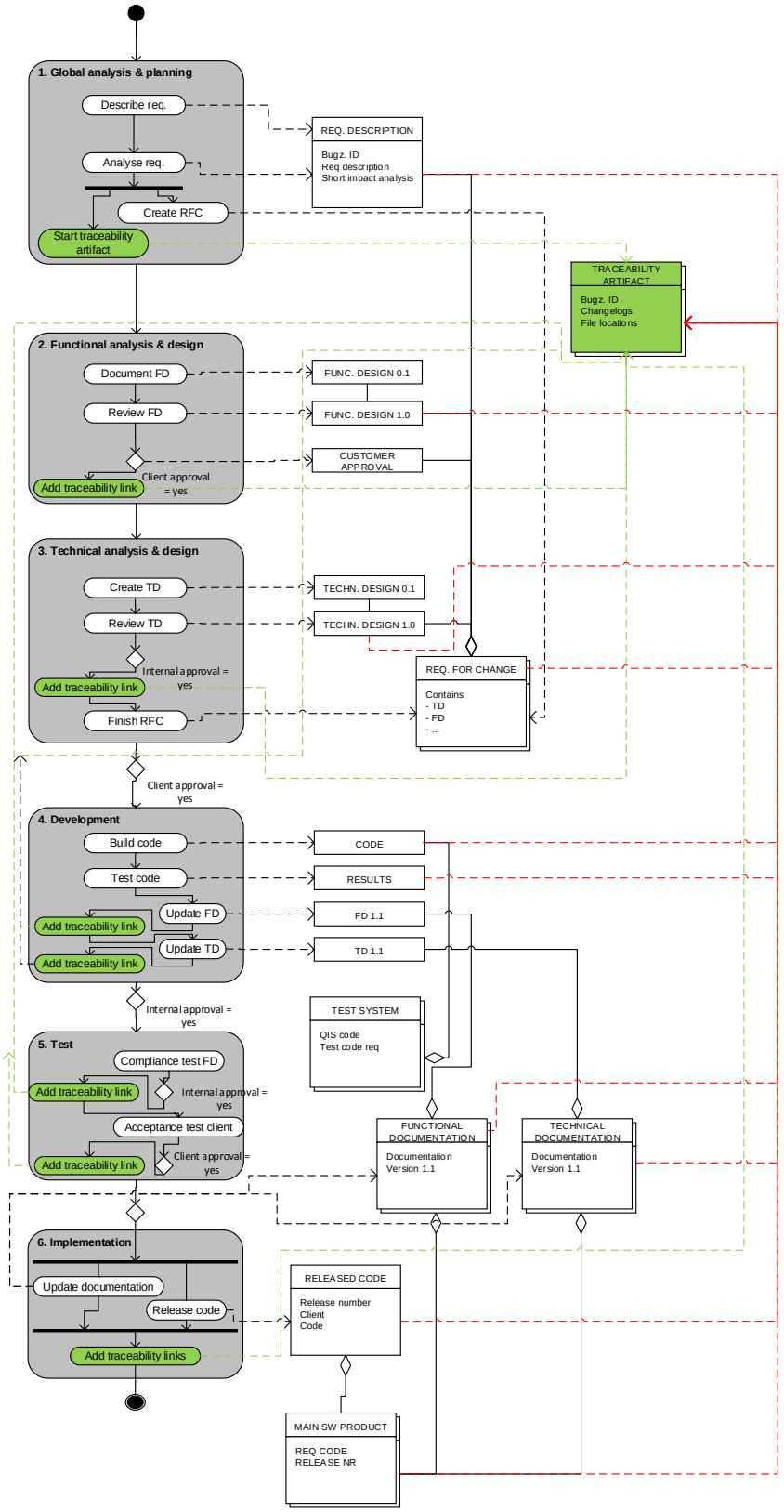



FIGURE 4.2: A PDD involving both Processes and Deliverables from the M1 Method.

# Chapter 5

# Experiment: Comparing M0 and M1

In Chapter 3, a series of interviews was done to find out the major problems in requirements engineering. In Chapter 4, the improved method M1 was introduced. This chapter shows the experiment that has been conducted for finalizing and concluding this study.

The type of this research is fixed design, implying that factors are fixed before the study is launched, also referred to as quantitative investigation. The explanatory research is mainly concerned with comparing two or more groups with the aim of identifying a cause-effect relationship [57].

## 5.1 Introduction to the Experiment

This experiment tries to answer the question if improving traceability also improves implementation effectiveness. This is done by asking a selected group of subjects to solve a a number of realistic assignments that require navigating through multiple documents. Some of the assignments are performed using method M0, and some using method M1. The subjects are not told which method they are using, to preserve validity. Each of the subject completes four assignments. Section 5.1.1 elaborates on this. Per subject, it is first asked:

- Which team he works
- What the function of the subject is

- How many years the subject has been employed
- How the subject would rate his working experience with respect to BugZilla (1 means none and 7 full).

For each of the assignments, next a number of variables was noted.

- Correct answer (yes/no)
- Time
- Perceived difficulty (1 means very easy and 7 means very difficult)
- Comments or Notes.

These variables will be stored and processed in Chapter 6. There, the results can tell if there is a difference in the variables when M0 is compared to M1. Section 5.1.1 explains the assignments that are used for this experiment.

### 5.1.1 Experiment Assignments

The data that is used for the assignments, consists of real-life data from actual bugs. This has been done for being representative for M0, while at the same time producing results that reflect regular results. In Chapter 4, the method M1 has been created which is M0 with full bi-directional traceability. The assignments, which are described later in this section, have been manually improved to full traceability. This has been done by copying the contents of the central traceability artifact to a local folder. As all assignments has been copied, there has been no signal for the subjects which assignments are from M0 or M1. Table 5.1 shows the different assignment numbers with bugnumber, direction, and traceability level. Note that from assignment 7, it is actually assignment 1 but with full traceability. Thus, 7-12 corresponds to 1-6. Figure 5.1 shows the traceability paths that are regularly available in both M0 and M1. Since a part of the assignments uses paths that are not present in M0 but are present in M1, the expectation is that added traceability paths will influence the variables mentioned in Section 5.1.

**Assignment 1** This assignment is based on a real bug. This bug, which is about the possibility of changing rules regarding the use of wildcards in a query, end with a comment saying that 'the document has been changed'. The objective here is to find which document has been changed, and what has been changed.

TABLE 5.1: BugZilla Numbers used in Experiment

| Assignment | Direction | Reference | Traceability |
|---|---|---|---|
| 1 | BugZ-Doc | 193268 | Average |
| 2 | BugZ-Doc | 214626 | Low |
| 3 | BugZ-Doc | 175174 | Low |
| 4 | BugZ-Doc | 221025 | OK |
| 5 | Doc-BugZ | 203493 | Low |
| 6 | Doc-BugZ | RFC 664 | OK |
| 7 | BugZ-Doc | 193268 | Full |
| 8 | BugZ-Doc | 214626 | Full |
| 9 | BugZ-Doc | 175174 | Full |
| 10 | BugZ-Doc | 221025 | Full |
| 11 | Doc-BugZ | 203493 | Full |
| 12 | Doc-BugZ | RFC 664 | Full |

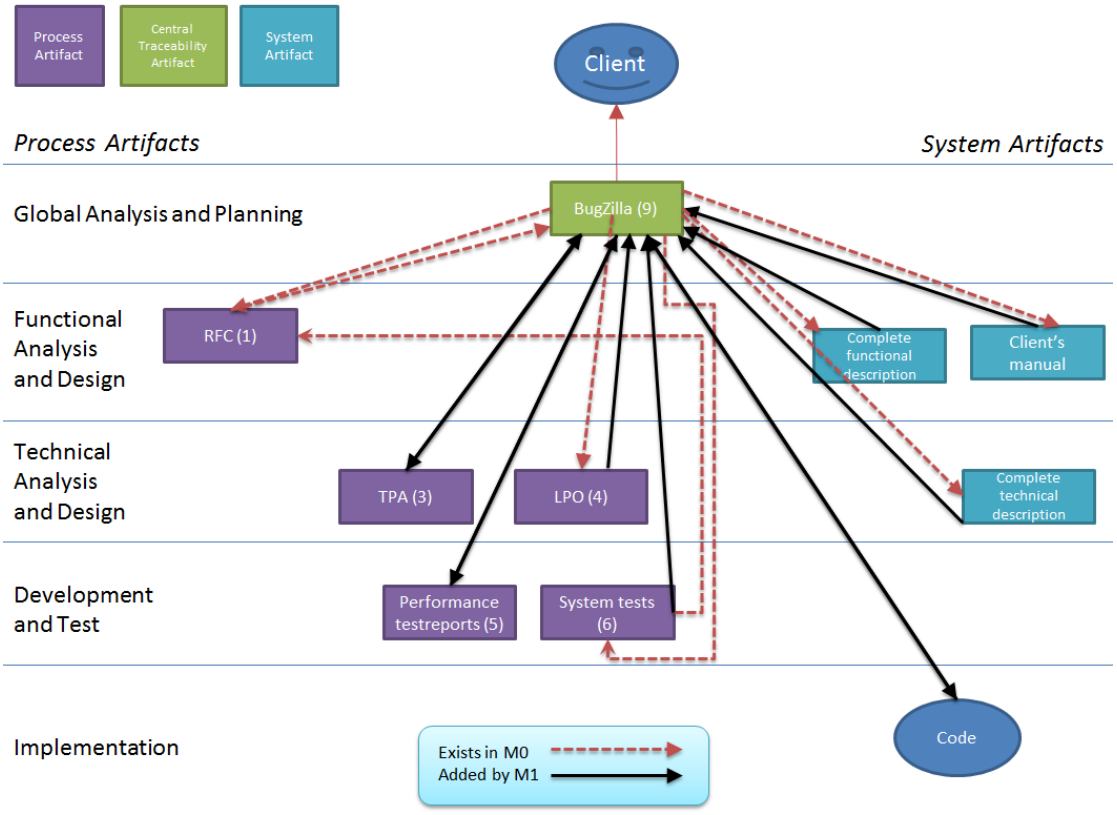



FIGURE 5.1: Traceability Links in M0 and M1

Table 5.2: Short- and Long Term Traceability Goals

| Category | Problem № | Short description | Fixed in short term? | Fixed in long term? |
|---|---|---|---|---|
| 2 | 10 | FD not clear enough | Allow access to other documents | Improvement by building KB |
| | 16 | Customer wish changes / FD not clear enough | Allow access to other documents | Improvement by building KB |
| | 18 | FD not clear enough | Allow access to other documents | Improvement by building KB |
| | 24 | FD not clear enough | Allow access to other documents | Improvement by building KB |
| | 25 | FD not clear enough | Allow access to other documents | Improvement by building KB |
| | 27 | Technical description not clear enough | Allow access to other documents | Improvement by better overview |
| | 28 | Tech. And func. designer don't understad each other | Allow access to other documents | Improvement by better overview |
| 3 | 7 | Client approves but later disagrees | - | Improvement by building KB |
| | 9 | Client approves but later disagrees | - | Improvement by building KB |
| | 12 | Client approves but later disagrees | - | Improvement by building KB |
| | 15 | Client approves but later disagrees | - | Improvement by building KB |
| | 21 | Client approves but later disagrees | - | Improvement by building KB |
| | 29 | FO is splitted and not tested against original | More horizontal traceability | More awareness of related requirements |
| 4 | 1 | Documentation hard to follow | Set pointers for clarity | Better estimation of impact |
| | 2 | Documentation hard to follow | Set pointers for clarity | Better estimation of impact |
| | 3 | Documentation hard to follow | Set pointers for clarity | Better estimation of impact |
| Cooperation | 20 | Teams don't talk enough | Allow access to other documents | Improvement by better understanding |
| Client | 11 | Client doesn't know what they want | - | - |
| Client | 13 | Client wants more and more functionality in same release | - | - |
| Client | 17 | Clients want to trust provider | Ability to hand over clear history | Better troubleshooting |
| Client | 26 | Client doesnt know he doesnt have knowledge | - | - |

*Legend:*
Category 1: Pre-RS RM
2: Incorrect FD
3: Incorrect FD approval
4: Documentation

> In this bug about the QIS search function, the new system settings are added in an FO. Can you find what exactly has been changed?

**Assignment 2** The second assignment is one with a very bad traceability. The BugZilla page only tells the user it is a 'dummy-bug'. When one knows where to look and what to search for, this bug is actually a bug for multiple clients. During this study, we were told that this is one of the typical examples of bad traceability.

> Which documents have been changed because of this bug?

**Assignment 3** For this assignment, the RFC document has to be found. There is a link to the file, but since the fileserver has migrated, the link is not working anymore. The objective here is to find the right document, and it needs some puzzling.

> For this bug, an RFC document has been created. Because of new bug, the RFC has to be checked. Can you give the file location?

**Assignment 4** This assignment consists of a BugZilla page with two posts. The first post contains a proposal for change, the second contains the text 'I made a change to the functional design document' at [file location]. The objective is to find out which changes were made. It needs reading of the previous post and comparing of documents to solve.

> Can you find the text that has been changed in FD 'Werken met [product]' by this bug?

**Assignment 5** This assignment starts with a document, and the objective is to go back to the original Bugzilla number. The document presented is an RFC document. The subject has to find out if the bug is approved or not.

> Has this bug been approved for building for the client?

**Assignment 6** This assignment provides a document that gives only one reference to the original bug: not the bug number, but the client's bug number. This means that some experience in finding a bug is needed to solve this assignment.

## 5.2 First Pilot Experiment

The first subject to do the assignments is a functional designer from the client team with only four months of experience. Although the time of experience is small, the subject rated a 4 on a 1-7 Likert scale for it, implying that a steady base of knowledge is already present. Table 5.4 shows the results.

After asking if there were any clear differences experienced between some of the questions, the reply was positive. The first two assignments took significantly more time than the others, making them easier. Moreover, it has been said that none of the questions was very hard. This is reflected by the score of the individual questions. Because this was the evaluation of the experiment, there was an open conversation later on about this experiment. The subject agreed that the choice of assignments could reflect requirements management. However, for the majority of the assignments, a less suitable source of data has been used. The majority of assignments, for that reason, had to be changed to better reflect M0.

Table 5.3: Planning of Subjects and Assignments

| Subject ↓ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 3 | | | 1 | | | | 2 | | | 4 | | 4 |
| 2 | | 1 | | | 2 | | | | 3 | | | 4 | 4 |
| 3 | | | 1 | | | 2 | 3 | | | 4 | | | 4 |
| 4 | 2 | | | 4 | | | | 1 | | | 3 | | 4 |
| 5 | | 3 | | | 1 | | | | 4 | | | 2 | 4 |
| 6 | | | 3 | | | 2 | 4 | | | 1 | | | 4 |
| 7 | 2 | | | 4 | | | | 1 | | | 3 | | 4 |
| 8 | | 3 | | | 2 | | | | 1 | | | 4 | 4 |
| 9 | | | 2 | | | 4 | 3 | | | 1 | | | 4 |
| 10 | 4 | | | 2 | | | | 1 | | | 3 | | 4 |
| 11 | | 2 | | | | | | | 1 | | | | 2 |
| Total | 4 | 4 | 3 | 4 | 3 | 3 | 3 | 4 | 4 | 3 | 4 | 3 | |

Table 5.4: Results of First Evaluation

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 1 | 4:30 | Y | 3 | "Never looked in this place" |
| 2 | 4 | 2:48 | Y | 3 | |
| 3 | 8 | 1:58 | Y | 2 | |
| 4 | 11 | 1:21 | Y | 2 | |

## 5.3 Second Pilot Experiment

The purpose of the second evaluation is to find out if the dataset better reflects the actual data produced by the team. The test went well, with no major remarks. However, the subject gave us the impression that she wanted to win, the tests were done as fast as possible. This interpretation of behavior is supported by the perceived difficulty of the assignments; all were rated 1 (Simplest). Afterwards, the subject told that the two assignments from M1 needed less effort, but didn't want to change the perceived difficulty (all valued 1 implies that they were the same difficulty). Table 5.5 shows the results of the second evaluation (note: assignments 1, 2, 4, 5, 6, 7, 8, 10, 11 and 12 were adapted to a situation that better reflects the M0 traceability situation as described in Section 5.2).

The subject remarked, after being explained what the setup of the experiment was, that there was a clear perceived difference between the M0 and M1 methods. In M1 it took less effort to find the documents. The second evaluation was successful.

## 5.4 Experiment Conduction

For performing the experiment, the desk of the author is used in a room in the main building of the case study organization. This is desirable for a number of reasons. First, since all settings and access policies are right for the experiment, it is the most logical choice, and second, by giving all subjects the same environment, the experiment is easier to validate. The subjects for the experiment were chosen to form a representative selection from the organization. This means, that from both client and non-client teams, both experienced and beginning individuals, a number were selected.

### 5.4.1 Subject 1

This experiment was performed with a subject which is employed as software engineer. He works in the client team, has half a year of working experience in this organization

TABLE 5.5: Results of Second Evaluation

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 1 | 1:57 | Y | 1 | "Never looked in this place" |
| 2 | 4 | 2:02 | Y | 1 | |
| 3 | 8 | 0:48 | Y | 1 | |
| 4 | 11 | 0:31 | Y | 1 | |

and rates his BugZilla expertise 4 on scale 1 (very litte) to 7 (very much). In the first assignment, in which the traceability in the M0 situation is relatively good, the subject did not trust the link. He did not try to click it, instead was used to search using other means. In the third assignment, the subject normally would have used another tool, which is unavailable in this experiment. Although he admitted at the end that traceability was likely to improve the method, he doubted if the organization would be able to maintain it.

### 5.4.2 Subject 2

This was done by a highly skilled member of the client team, having nine years of experiment starting as software engineer and later on working as tester and teamleader, too. His understanding of Bugzilla was between 6 and 7 on a 1-7 scale, which is a lot. Table 5.7 shows the results. Although the first assignment was already known to the subject, it seemed impossible to find all the related documents. After almost seven minutes he gave up. The third assignment was interesting, too. Although an easy clickable traceability link was added, the subject did not see it and found the good answer in another manner. When asked if he experienced differences, he could tell immediately that two assignments were much easier to solve than the others. After explaining what this study really did, his opinions did not change. He added that the addition of traceability was a very good idea and it really helped in easing findability of related documents.

### 5.4.3 Subject 3

The third subject is an experienced employee, who has been contracted for seven years. His experience in BugZilla was rated a 5 on a 1 to 7 Likert scale. His team works with the main product, not specifically for one client. His function is leader of the team. Table 5.8 shows the results of this experiment. In the first assignment, the subject was navigating back to the original bug. From there, he managed to find the answer to the

Table 5.6: Results of Subject 1

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 4 | 3:14 | Y | 2 | "Subject did not trust traceability" |
| 2 | 8 | 0:28 | Y | 1 | |
| 3 | 1 | 4:12 | N | 3 | "Would normally use SVN[1]" |
| 4 | 11 | 1:15 | Y | 3 | |

assignment. This is interesting, because this possibility was not known. When doing the second assignment, the subject did not even open the BugZilla page, he knew by heart where the files were. The same happened with assignment three and four. The fourth assignment was answered correctly; however, the remark was that although the change has been adopted in the document, there has not yet been a test. So the document has changed but it has not been sent to the client. The subject told the next remarks after explaining what this experiment is about: It would be nice if the links to files actually point to the files instead of being faulty. He added, that specifically for clients with access to BugZilla it can be very hard to track a bug/requirement.

### 5.4.4 Subject 4

The fourth subject is an experienced software engineer. He is working at a client team for two years, and scales his experience with BugZilla on a 4. The first assignment, number 8, passed relatively fast, with 75 seconds. This was assignment the traceability enabled version of assignment 2. All the traceability links to related documents are in the BugZilla page. The subject said that finding the documents was very easy, however if he was to validate all documents including changes, it might have taken more time. It took the subjects about half a minute to verify that the document indeed was changed. The second assignment, being number 1, took 2 minutes and 10 seconds. The subject found the answer to the experiment without opening the document; however he was

Table 5.7: Results of Subject 2

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 2 | 6:44 | N | 6 | "Would normally open mailbox and send email. Knows the bug. Tried other possibilities" |
| 2 | 5 | 1:40 | Y | 3 | "Perform advanced search using bug number" |
| 3 | 9 | 3:36 | Y | 2 | "Used advanced search instead of easy link" |
| 4 | 12 | 0:24 | Y | 1 | "Easy" |

Table 5.8: Results of Subject 3

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 3 | 1:53 | N | 2 | |
| 2 | 6 | 0:49 | Y | 2 | "Didn't have to open file" |
| 3 | 7 | 0:50 | Y | 1 | |
| 4 | 10 | 0:28 | Y | 4 | "Not yet tested for approval" |

advised by us to find the document for verifying the answer. The third assignment, originally number 4, was solved in 1 minute and 13 seconds. The last one, being number 11, was initially answered wrong. The case is, that the BugZilla page for this assignment has multiple posts about the approval, which is the objective of the assignment to find. Since the first post told that there was an approval, the subject thought the bug was approved. However, after looking down, he corrected the answer, as a number of posts later there is another post correcting the approval tot non-approved. Table 5.9 shows the details of experiment 4.

### 5.4.5 Subject 5

The fifth subject is a beginning software engineer. He has about three quarters of a year experience. His experience in BugZilla is rated 5. He works for a client team. For the first assignment, the subject used the RFC code of the document for finding the original BugZilla entry. For the second, after he found the right answer, he kept searching; later he explained me that he thought the assignment couldn't be that simple. The time in which he realized that he found the answer has been noted. The third assignment proved to be very hard; the poor traceability combined with a difficult bug, makes it hard to find related documents. After 1 minute 50 seconds the subject gave up. The fourth assignment included a tracebility link (clickable), but because a large number of links don't work since the fileserver migration, the subject did a manual search instead of even trying the link. Again, the time needed for the subject to realize that he has the right answer, instead of the time that it actually took to find the right answer, has been noted for this assignment. At the end of the assignment, the subject told that he thinks that traceability is essential for navigating through the BugZilla-'forest'. Table 5.10 shows the details of assignment 5.

Table 5.9: Results of Subject 4

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 8 | 1:15 | Y | 1 | |
| 2 | 1 | 2:10 | Y | 2 | |
| 3 | 4 | 1:13 | Y | 3,5 | |
| 4 | 11 | 1:04 | Y | 5 | "Hard to find, because posts cannot be corrected" |

### 5.4.6 Subject 6

The experiment with subject six was performed with a software engineer for a client team. He has been employed for 2,5 years, and rates his experience 6 on 1 (very little) to 7 (very much). He has spend a large part of his time working with BugZilla, and thus has a lot of experience finding the unfindable requirement related data. In the first assignment, an easy link was included by the authors, since it is part of the M1 method. He did not click on the traceability link however, and later on explained that 'direct traceability never really works' in the current system. So, he did not try to use it. The time for solving the assignment using the M0 method has been used instead. The second assignment was hard according to the subject, because it is a big bug, where the traceability is missing. With some more searching however, he succeeded in finding the answer. The third and fourth assignments didn't pose any problems. Table 5.11 shows the findings of this experiment. At the end, he had the remark that in order for traceability to work, surely the employees will have to be notified. This avoids situations where the traceability is good but is not used because of the situations in which it does not work. Moreover, the subject told that M1 is a greatly improved version over M0.

TABLE 5.10: Results of Subject 5

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 5 | 1:23 | Y | 3 | |
| 2 | 12 | 1:38 | Y | 2 | "Did extra check to confirm that he had the right answer" |
| 3 | 2 | 1:50 | N | 7 | |
| 4 | 9 | 1:28 | Y | 3 | "Found answer without using traceability" |

TABLE 5.11: Results of Subject 6

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 10 | 1:43 | Y | 2 | "Did not use M1 functionality" |
| 2 | 6 | 2:16 | Y | 4 | "Hard because the complexity combined with bad traceability" |
| 3 | 3 | 0:54 | Y | 2 | |
| 4 | 7 | 1:03 | Y | 1 | |

### 5.4.7 Subject 7

This experiment has been conducted with help of a functional designer from the main product team. He has been contracted at the case study company for three and a half years. His experience with BugZilla was rated a 5. The first assignment was very easy, it was solved in only 11 seconds and difficulty was rated 1. In the next assignment, the subject was looking for the document. He picked a document, which unfortunately was the wrong one: looking in that document, the subject realized that the paragraph does not exist. His answer thus is incorrectly. It took about three minutes to find this out, and the question was rated a 3 on difficulty. The assignment afterwards took about three minutes, but was also answered wrongly. It was rated a 2 for difficulty. The last assignment was answered correctly in about one minute, rated a 2 for difficulty. At the end, after explaining where the experiment is about, the subject added that he liked the traceability, and that he would welcome and use it.

### 5.4.8 Subject 8

This experiment has been conducted with help of a senior functional designer. The subject has been employed for 5 years in this organization. The expertise in BugZilla has been rated a 3. The first assignment was finished fairly easy in just over half a minute. The subject complained that the traceability link only appeared after reading quite some posts (note that this is one of the easiest assignments). The second assignment was not performed well. After a period of only only 43 seconds, the subject gave up and told that at this moment it would cost too much of his time. He would call the one responsible. The third assignment was remarkable as well; as this one is considered the hardest by the previous subjects, in this experiment the subject solved the assignment. After initially giving up, the subject kept trying, and after a few minutes more, he found a document with traceability-links to other involved documents. The last assignment was solved without any remarks. Table 5.13

Table 5.12: Results of Subject 7

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 8 | 0:11 | Y | 1 | |
| 2 | 1 | 3:03 | N | 3 | |
| 3 | 4 | 2:48 | N | 2 | |
| 4 | 11 | 0:58 | Y | 2 | |

### 5.4.9 Subject 9

This experiment was conducted with a software engineer from the main product team. He has been employed for 1,5 years, and would rate his experience in BugZilla a 6 out of 7. The first assignment was performed in just over two minutes, with correct answer and hardness rated 3 out of 7. The second assignment was initially quited after about three and a half minutes, however the subject remembered another search function and asked if he could continue. This was allowed, and the extra time was added to the original time. This resulted in almost four minutes for completing the assignment correctly. This assignment was rated a 6 on difficulty. The third assignment was solved in about one and a half minute, the subject remarked that the description was too generic. "Why don't they tell what has changed, instead of saying that there is a change", was among a few remarks. It has been solved correctly and is rated a 3 on difficulty. The last assignment was solved in about four minutes. The answer changed three times, the last one being correct. The subject remarked "Oh no.." several times, when navigating through the bugs and documentation. Difficulty was rated a seven.

Table 5.13: Results of Subject 8

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 9 | 0:37 | Y | 2 | "This assignment should not be that hard" |
| 2 | 5 | 0:43 | N | [none] | "I quitted, assignment too hard" |
| 3 | 2 | 4:44 | Y | 6 | "Fileserver migration makes life hard" |
| 4 | 12 | 0:26 | Y | 2 | |

Table 5.14: Results of Subject 9

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 10 | 2:01 | Y | 3 | |
| 2 | 3 | 3:47 | Y | 6 | "First gave up but continued looking, found answer 20 sec later" |
| 3 | 7 | 1:33 | Y | 3 | "Would be nice if the descriptions were less generic" |
| 4 | 6 | 3:51 | Y | 7 | "A lot of clicking for this bug" |

### 5.4.10 Subject 10

The tenth experiment was performed with an unexperienced software engineer. He is employed for 8 months and rates his experience in BugZilla a 3. The first experiment has been solved quickly, correct and is rated a 1 for difficulty. The second was rated a 5, needed about four and a half minute but has been done correctly. The third assignment took about one minute, has been done correctly and rated a 3. The last one was very hard for the subject; since there was little traceability, he could not find the file needed for answering the question. It took the subject about four minutes to give up.

### 5.4.11 Subject 11

This experiment has been conducted with a software engineer that has been employed for three and a half years. He rates his experience in BugZilla at 6 out of 7. The first assignment was finished fast, with only sixteen seconds needed. It was rated a 1 on difficulty, on a 1-7 scale. The second was answered correctly in about three and a half minute. This is impressive, when compared to the rest of the participants' scores at this assignment. Via a document describing this requirement, the other documents were found that have been adapted as result of this requirement. The third assignment has been performed in under half a minute with good result, the fourth in just over one minute with good result.

Table 5.15: Results of Subject 10

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 8 | 0:23 | Y | 1 | |
| 2 | 4 | 4:24 | Y | 5 | |
| 3 | 11 | 0:59 | Y | 3 | |
| 4 | 1 | 4:03 | N | [none] | "Structure of the new file-server is confusing" |

Table 5.16: Results of Subject 11

| № of ass. | № of real ass. | Time | Correct | Difficulty | Remarks of the subject |
|---|---|---|---|---|---|
| 1 | 9 | 0:16 | Y | 1 | |
| 2 | 2 | 3:22 | Y | 2 | "Using the RFC document the other documents were found" |
| 3 | 12 | 0:23 | Y | 1 | |
| 4 | 5 | 1:13 | Y | 1 | |

### 5.4.12 Conclusion

The experiment went according to the plans and no anomalies appeared. During the experiment, we had the impression that the traceability-enabled method M1 is easier for most subjects to perform. Chapter 6 will elaborate on the these results.

# Chapter 6

# Analysis of the Results

After showing some tables presenting general results, the following sections report on presenting general results. From Section 6.1 on, the deeper analysis is performed, studying the effect of experience on traceability improvement. Since the assingments from the M1 experiment did not produce incorrect solutions while those at M0 did, we define M0✗ as the M0 method including the incorrect answers and M0✓ as the M0 method not including the incorrect answers. Table 6.1 shows the average scores for both methods. The M1 averages show a decrease in both time, difficulty and standard deviation of time. Note that in some cases it was not possible to exclude variables for wrong answers, as the amount of variables would be too low. In the case of three variables, the boxplot can only serve as indication, and is marked with the § sign. According to the Shapiro Wilk Normality Test, the time scores are not normally distributed [58].

## 6.1 General Analysis

In Table 6.1, the average time is presented, as well as the percentage of correct answers, perceived difficulty and the standard deviation of time. It is presented for M0✗ (the M0 method that includes times for wrong answers) M0✓ (for M0 only including the time needed for right answers) and M1 (with the ideal traceability). It shows that there is a clear difference of more than 60% between the average time for M0 and M1. The

TABLE 6.1: Average Results from Experiment.

| Method | Time (S) | Correct (%) | Difficulty (1-7) | St. Dev. of Time (s) |
|---|---|---|---|---|
| M0✗ | 162 | 71 | 3.71 | 93.04 |
| M0✓ | 152 | 100 | 3.54 | 79.55 |
| M1 | 63 | 100 | 2.13 | 46.94 |

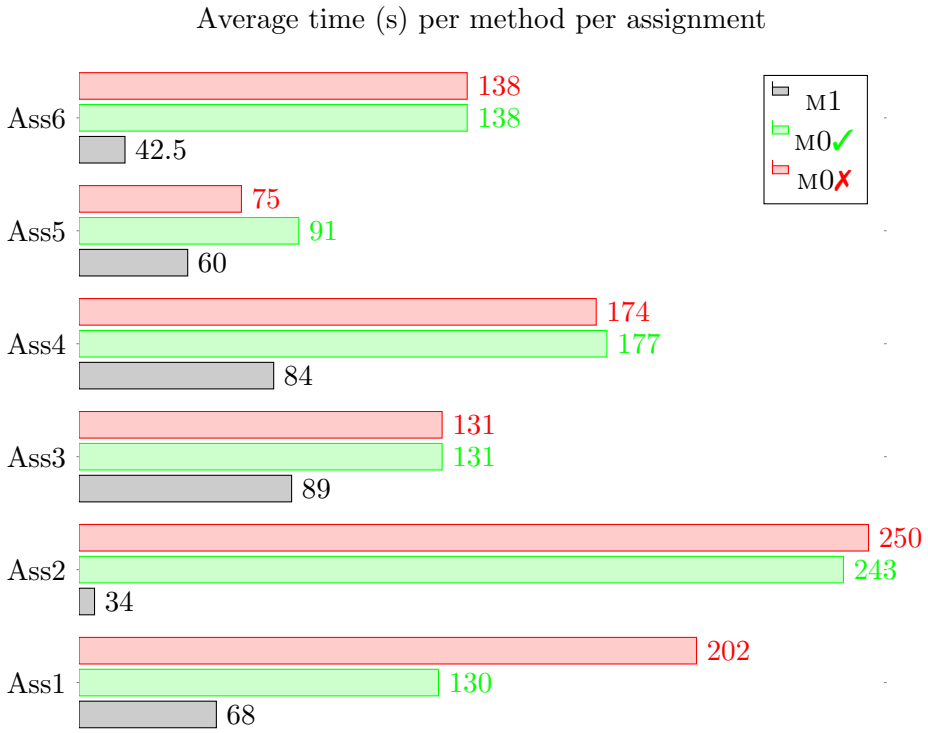



FIGURE 6.1: Average Time per Method per Assignment

number for correctness in M0 is 71, while in M1 it is 100. The average of difficulty is lower in M1, because of the ordinal character, it will be analyzed later in this chapter. The standard deviation of the distribution of time looks very different when compared between M0✓ and M1. In Figure 6.1, the average time per assignment is shown in a bar chart. The figure visualizes the time for an easier understanding. It is interesting to see that different assignments yield different results. If the M0✗ is compared with M0✓, the mean time needed for completing assignment 1 dropped from 202 to only 130 seconds. This happens if a participant provides the wrong answer and needs a relatively large amount of time. Note that it is not always the case; in assignment 4, one of the participants was sure about the correctness of his answer in a short time. There, the mean time rose from 174 to 177 seconds by eliminating the times for wrong answers. When comparing M0 and M1, assignment 2 has the highest difference. It is also the hardest assignment to solve, with the classification of low traceability. But overall, the results show that in comparing averages, clear improvements in time can be seen, both with M0 including wrong and excluding wrong answers. Section 6.2 will continue analyzing the distribution of times.

TABLE 6.2: Variances in Times for Figure 6.1

| Assignment | Method | Variance |
|---|---|---|
| 1 | M1 | 250.7 |
| | M0✓ | 0 |
| | M0✗ | 2431.5 |
| 2 | M1 | 591.7 |
| | M0✓ | 1681 |
| | M0✗ | 11694 |
| 3 | M1 | 6040.7 |
| | M0✓ | 5156.2 |
| | M0✗ | 5156.2 |
| 4 | M1 | 1291.5 |
| | M0✓ | 3660.3 |
| | M0✗ | 4683.7 |
| 5 | M1 | 5.3 |
| | M0✓ | 72.3 |
| | M0✗ | 558.2 |
| 6 | M1 | 1184.9 |
| | M0✓ | 5617.3 |
| | M0✗ | 5617.3 |

### Correlation Analysis

A two-sided correlation matrix has been created for both M0✗, M0✓ and M1. Because of the non-normal distributed character of time needed for M0, it is not recommended to use Pearson's R. For this reason, Spearman's $\rho$ is used. Table 6.3 shows the correlation matrix for the results of M0✗. The terminology used for describing the correlations, is proposed by Evans in [59]. It is shown in Table 6.4.

Between difficulty and time, a moderate positive correlation can be identified. This is explainable, because assignments that take more time to solve, require more effort, which translates in a higher difficulty. Between *BzExpert*, which should be read as BugZilla Expertise, and *YearsExp*, which stands for years of experience, there is a moderate correlation, too. This can be explained because more years of experience working at the

TABLE 6.3: Spearman's $\rho$ Correlations Matrix of M0✗

| | T$_{(seconds)}$ | Difficulty | Correctness | YearsExp | BzExpert |
|---|---|---|---|---|---|
| S | 1 | | | | |
| Difficulty | **0.52** | 1 | | | |
| Correctness | -0.35 | -0.16 | 1 | | |
| YearsExp | 0.07 | -0.1 | -0.04 | 1 | |
| BzExpert | -0.08 | 0.05 | -0.06 | 0.41 | 1 |

TABLE 6.4: Verbal Descriptions of Correlation Coefficient by Evans in [59].
*N.B.: the verbal descriptions count for both positive and negative values of $\rho$.*

| Spearman's $\rho$ | Verbal Description |
|---|---|
| 0.00 – 0.19 | Very weak |
| 0.20 – 0.39 | Weak |
| 0.40 – 0.59 | Moderate |
| 0.60 – 0.79 | Strong |
| 0.80 – 1.00 | Very strong |

case study organization will increase the level of BugZilla experience. Between *Correctness* and *Time*, there is a weak negative correlation. This implies that an assignment that takes more time, is more likely to be answered wrongly. Table 6.5 continues with showing correlations for M0✓. It is interesting that the correlation between *Difficulty* and *Time* increased to a strong one, which may be explained by excluding the noise that is introduced by including the wrong answered questions. *YearsExp* now has a weak negative correlation with *Time*, which implies that more years of experience lead to higher efficiency in solving the assignments. This is in line with the impression that the authors got during the experiment. The correlation between *BzExpert* and *Time* is interesting too, being weak and negative, it tells that more experienced participants require less time to solve the questions generally. The correlations for M1can be found in Table 6.6. When compared to Table 6.5, the correlation between *BzExpert* and *Time* changed the most; from a weak negative correlation of $-0.23$, it rose to a positive weak correlation of 0.33. An explanation for this change is that M1 decreases the disadvantage of being inexperienced; when the participant is more experienced, the amount of time for solving the assignments even increases, which implies that the inexperienced participants are now faster in solving assignments than the experienced ones.

TABLE 6.5: Spearman's $\rho$ Correlation Matrix of M0✓

| | T$_{(seconds)}$ | Difficulty | YearsExp | BzExpert |
|---|---|---|---|---|
| S | 1 | | | |
| Difficulty | **0.61** | 1 | | |
| YearsExp | -0.34 | -0.29 | 1 | |
| BzExpert | -0.23 | -0.02 | 0.35 | 1 |

TABLE 6.6: Spearman's $\rho$ Correlation Matrix of M1

| | T$_{(seconds)}$ | Difficulty | YearsExp | BzExpert |
|---|---|---|---|---|
| S | 1 | | | |
| Difficulty | **0.47** | 1 | | |
| YearsExp | -0.2 | -0.15 | 1 | |
| BzExpert | 0.33 | -0.1 | 0.41 | 1 |

## 6.2 Time Distribution Analyses

In this section, the distribution of time needed for solving the assignments is represented by a number of boxplots, because of the representative properties. First, Figure 6.2 shows the general distributions of time. The top two boxplots show the distribution of time scores among all experiments, both in M0 and M1. Below, the boxplots for M1, M0✓ and M0✗ are shown. In this manner, the different distributions can be compared to the general distribution; both methods have a wide spectrum of time scores, but in M1 the middle 50 percent of scores is more centered than in M0. Next, the analysis is performed per traceability class. In [6], it is said that both forwards and backwards traceability are important. However, since the central node (BugZilla) is used more commonly than a related document, in the forwards traceability (central node → related information) three different classes of traceability are created, where in backwards traceability (related information → central node) only one has been tested. The time results are classified as follows[1]:

(i) Low versus full forwards traceability

(ii) Average versus full forwards traceability

(iii) Good versus full forwards traceability

(iv) Average versus full backwards traceability.

**Low versus full forwards traceability** If the traceability is low, there are little traces to related information. As the room for improvement is large here, this category is expected to benefit the most from improving traceability. It is shown in Figure 6.3. This figure shows, that in low traceability, the median in M0 is about four times the value of M1. About 60 percent of the times from M1 are outside of the M0 distribution. This shows that in low traceability, there is a large difference between the times.

**Average versus full forwards traceability** Moderate traceability links to a related document, however the link is wrong. Using the syntax of the link, it is possible to find out where the documents are, however experience is needed to be able to do this. Figure 6.4 shows the results for the average forwards traceability assignments. The distribution of M0 has more overlap with M1.

[1]The amount of data that was gathered for this analysis is too small in some occurrences. For this reason, it was not possible to plot Figure 6.3, Figure 6.4 and Figure 6.5 with only M1✓. Since this is a threat to validity, it is added to Chapter 7.

**Good versus full forwards traceability** This category is closest to the full traceability situation. Sometimes details are missing, but in general, the traceability is sufficient for finding related data. Figure 6.5 shows the distribution of time for solving assignments in this category. Please note the § sign, which means that there was not enough data for the boxplot to be representative. For presenting an indication, the M1 boxplot was included. If the change in results from the low and average traceability would consist in this category too, the expectation for the results was that there would be less differences than in the average category. This M0 data however, is very similar to Figure 6.4.

**Average versus full backwards traceability** In order to see what effect improving traceability has on backwards traceability, two assignments have been created to test

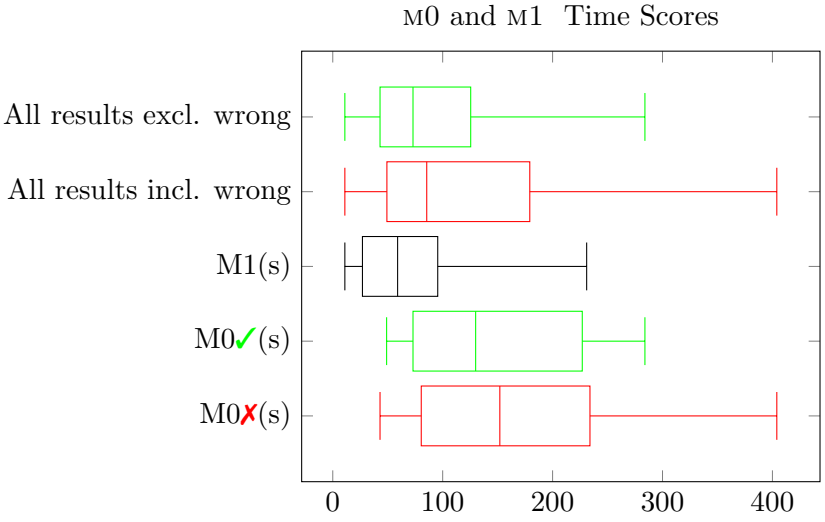



FIGURE 6.2: Boxplot Plotting M0 and M1

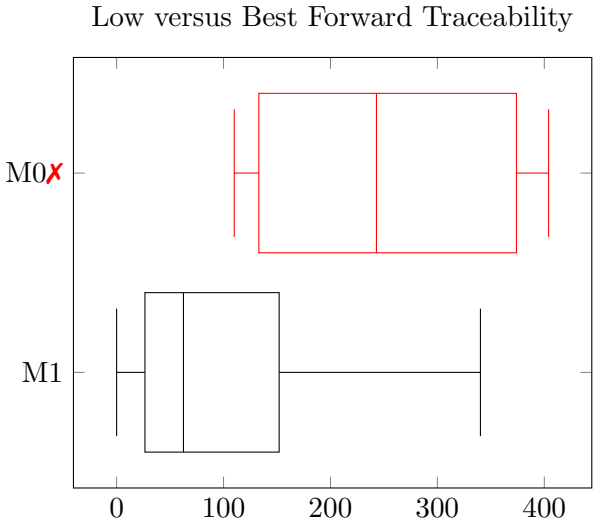



FIGURE 6.3: Boxplot Plotting Times of Low and Best Forwards Traceability

this. Number five ask the participant to find an original bug number in BugZilla, for which the traceability route has to be known. It is traceable, if one knows how to navigate through BugZilla. Assignment six is also classified as having moderate backwards traceability. The participant has to find the original bug, where the answer can be found. The distributions of this classification of traceability times can be found in Figure 6.6. Both M0✗ and M0✓ are largely different to M1.

## 6.3 Difficulty Rating Analysis

If it is easier to perform a task, it can result in higher efficiency and/or effectiveness. However, other confounding variables can influence the results of the experiment. To

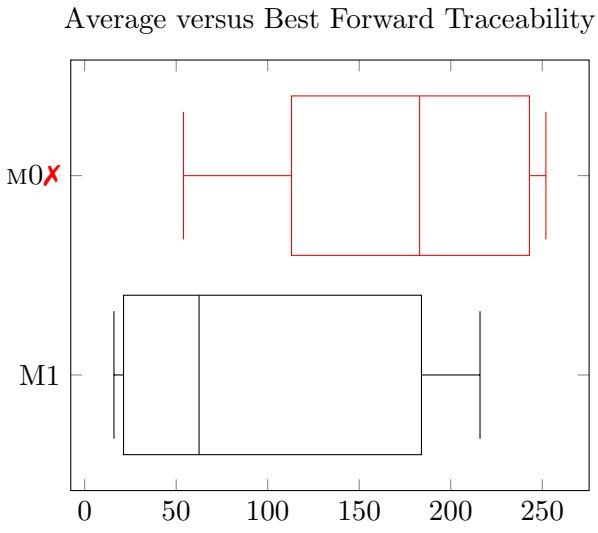



FIGURE 6.4: Boxplot Plotting Times of Average and Best Forwards Traceability

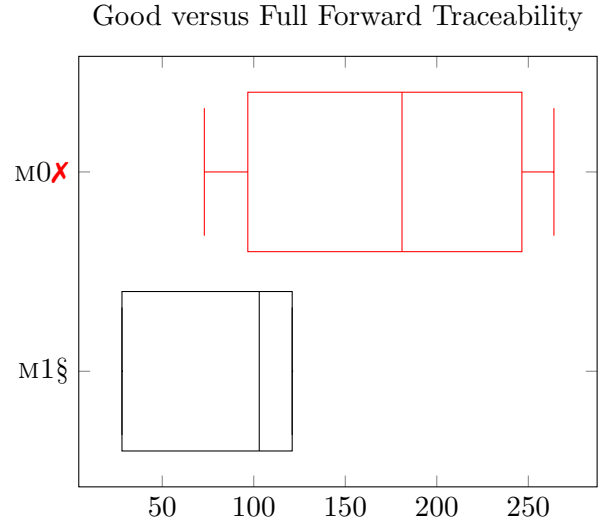



FIGURE 6.5: Boxplot Plotting Times of Good and Best Forwards Traceability

support the conclusion of this study, the participants were asked to rate the difficulty for each question. It is expected that, since time differs between M0 and M1, a difference will be seen in rating of difficulty too. Figure 6.7 shows the differences in ratings on difficulty between all results, M1 and M0✗. It is clearly visible that M1 is rated more towards strongly agreeing than towards strongly disagreeing. For M0✗, it is the opposite. This means, that the results in terms of difficulty ratings support the results regarding time.

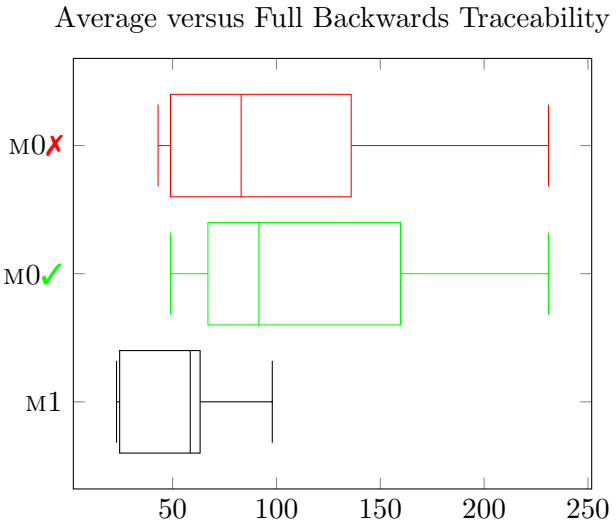



FIGURE 6.6: Boxplot Plotting Times of Average and Best Backwards Traceability

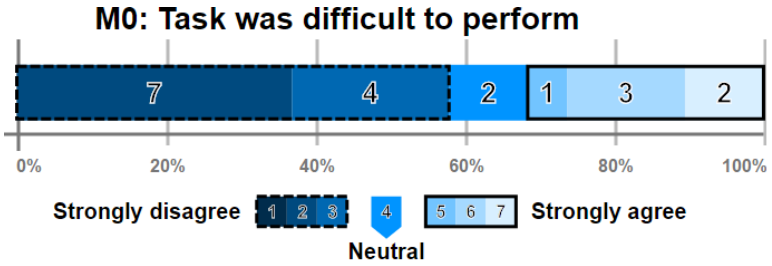



FIGURE 6.7: Distribution of M0 Difficulty Ratings

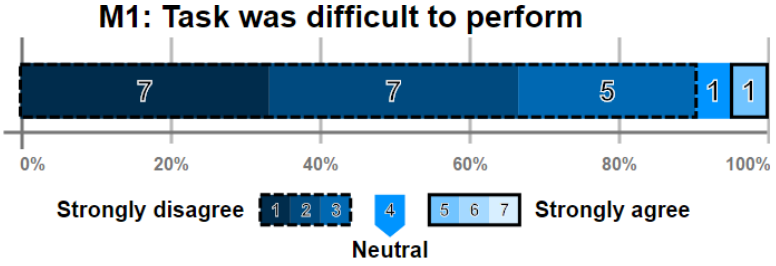



FIGURE 6.8: Distribution of M1 Difficulty Ratings

## 6.4 Conclusion

Most results are showing differences between the low-traceability method M0 and the traceability-enabled method M1. This accounts for the time that is needed for solving different assignments, as well as the perceived difficulty of solving the assignments. Moreover, as showed in Table 6.5 and Table 6.6, the correlation between the perceived difficulty and the amount of time needed to solve the assignment, dropped from a strong positive .61 to a moderate positive .47. Although the dataset is small, it can be caused by the statement that a more difficult task in the M1 method increases the needed time less than it would in the low-traceability M0 method. Another interesting correlation is the self-rated experience in using this system versus the time in seconds. Where in method M0 the more experience does not necessarily lead to a faster solution with a weak negative correlation of -.23, in method M1 it does, having a weak positive correlation of .33. Common sense would say that more experience leads to more efficient working; which does not seem to be the case. A possible explanation is that the more experienced subjects already know how to find the information, while not looking for clues in traceability. Behavior during the experiment in Chapter 5 supports this theory; some of the more experienced subjects did not use the traceability links, even when they noticed the links. Chapter 7 reports on the threats to validity, and Chapter 8 concludes the study.

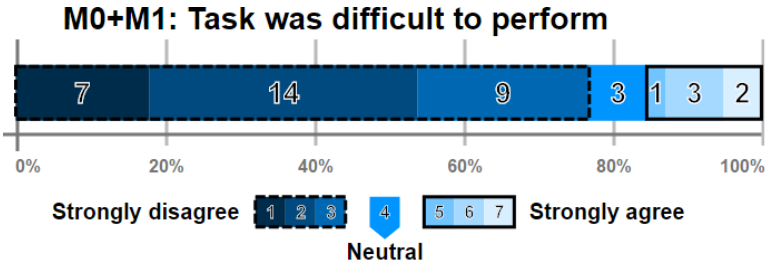



FIGURE 6.9: Distribution of Combined M0 and M1 Difficulty Ratings

# Chapter 7

# Threats & Validity

In this section, the different types of threats to validity will be discussed.

## 7.1 Internal Validity

The internal validity refers to the causality between two variables. The procedures are selected to make sure that there is as little as possible of unwanted influences on the final data. However; it can not be fully guaranteed due to the complexity and the human factor in this experiment. The measures that are taken to ensure a high internal validity are: (i) Using the same source data and assignments in both methods (ii) Use of same procedure for performing the first and second measurement and (iii) the use of the same subjects in both measurements.

### 7.1.1 Causal Relationship

Due to the available resources for this study, it unfortunately is not possible to prove that improving traceability increases implementation effectiveness; only an experiment was possible if not too much of resources (time) were used. For this reason, not a large amount of data was used for analysis. That is why we can only say it is likely or not likely to improve. However, if the case study organization decides to adopt the M1 method, it may be possible to compare the effectiveness of both methods in real life production environment.

### 7.1.2 Confounding Variable Influences

Because the setting for the experiment including the assignments is not completely the same as a normal working environment, it may cause confounding variables to be introduced. Since both time, score and difficulty all point to the same conclusion, and the only variable that changed was traceability, we assume the impact of confounding variables little to none.

### 7.1.3 Domain Knowledge

To eliminate the confounding variable of domain knowledge (we must measure what we want to measure, and this checks if we measure that), the experiment will be validated by an expert and a beginner in the domain. This validation is expected to raise the result that more domain knowledge leads to a better score in M0, minimizing the difference between M0 and M1. The control test is the beginner, for which it is expected that little domain knowledge leads to a greater difference in results between M0 and M1.

### 7.1.4 Evaluation Apprehension

The authors are watching the subjects carefully while conducting the experiment. If there is any indication of this phenomenon (subjects that perform poorly because of the fear of being evaluated), the results cannot be regarded as treatment effect. Moreover, by letting all subjects perform two M0 and two M1 assignments in an unknown order, there is a small change of changing the behaviour to satisfy the researchers. Moreover, the subjects were told that speed is not important, they should take as much time as they usually do.

### 7.1.5 Experiment Knowledge & Hypothesis Guessing

The subjects are not told what the exact purpose of the experiment is; they are told that this experiment is being conducted to find out how different bugs are 'solved' by the subjects.

### 7.1.6 Method Knowledge

For checking the influence of M0method knowledge on the difference in results, all test subjects are asked what their job is and how much experience they have.

### 7.1.7 Pressure

Since the experiment is anonymized, there is no possibility of tracing back the results. The subjects are noticed about this. Since the time is measured during the assignments, the subjects are told to perform the assignments at the same speed it is normally done. Since both the M0 and the M1 assignments are performed in the same setting and the subjects don't know which is which, the effect should be the same on all durations.

### 7.1.8 Easiness of Performing Assignments in M1

A threat to the validity of this study may be the ease of which M1 tasks were performed. Since the tasks were all performed right, this could mean that the tasks actually were too easy. The authors tried to avoid that by only picking daily tasks, however. Changing tasks between M0 and M1 would have been significantly more harmful to the validity.

## 7.2 External Validity

The external validity refers to the generalizability of the observed causal connection. In principle, the external validity is high.

### 7.2.1 Generalizability

This study has been applied to a medium-sized insurance software development organization that uses agile software development methods. Because software development is relatively general, the results from the experiment are likely to be generalizable to other fields of software development. It is important to note however, that the tool used for requirements management can limit the generalizability; the experiment showed that from low to ideal traceability there was more improvement than from the moderate or OK traceability to ideal traceability. The results should be generalizable to a large number of software product development organizations, which fit in the following collection of attributes:

- Requirements Management currently fits in the *3. Structured* level of the RMMI
- The software development method has agile characteristics
- The development team exists out of a number of people similar to the case study

Future research should point out whether the results are more generalizable, either:

- Beyond the broadness of agile software development
- Beyond the size of development teams
- Outside of the insurance software development domain

### 7.2.2 Selection Bias

The test subjects have not been selected randomly, because of the available resources. If, for example, the distribution of subjects in the different teams is not even, the results from this study may not be generalizable to all of the company. Moreover, the selection of test-subjects was not random: as the junior employees are relatively new, they are relatively easy in making time for experiments and interviews. The senior employees were harder to interview, because of their full schedules. Next, the setting of the experiment was such that it represents a working environment, but since the participants were not allowed to use other methods than BugZilla for tracing requirements, the results from the experiment do not fully represent the actual situation. (For example; if a participant would have called another one that knows the answer, he would score better on time.) Due to the nature of this study, the resources are limited. Since employees in the case study organization are busy, all applicants were accepted for the test. This means, that in practice, more software engineers than functional designers, team leaders and testers have been interviewed.

## 7.3 Conclusion

Although a number of threats to validity have been identified; to the understanding of the authors, the study has been performed as valid as possible. As stated before, unfortunately the resources were the main limit for this study.

# Chapter 8

# Conclusion

In Chapter 1, the main research question was posed. For readability, it is stated again:

> *Research question*
>
> To what extent does improving requirements traceability also improve the implementation effectiveness in the product software domain?

Figure 8.1 illustrates how the research question is answered. The causes for low requirement implementation effectiveness are identified. The red square illustrates which causes are mitigated by improving traceability. After that, an experiment will tell how improving traceability effects the cause.

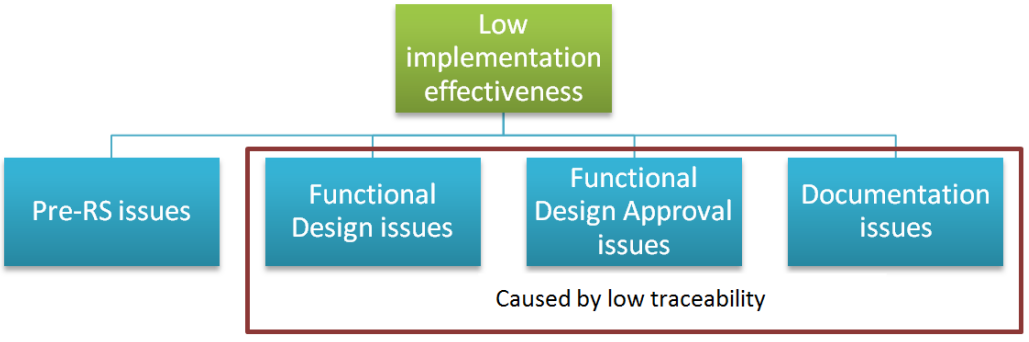



Figure 8.1: Research Question Explained

## 8.1 Is the Low Implementation Effectiveness caused by Low Traceability?

First, the question was answered what causes for low requirements implementation effectiveness can be mitigated by improving traceability. The interviews pointed out that most of the relevant problems arose because of the difficulty of finding related information regarding a requirement or an implementation, or related requirements. When the needed information is not easy findable, it causes employees to skip the search for that information instead of finding the right information in documentation. The first option often proves to be wrong. Since the definition for requirements traceability from [6] says that traceability refers to the ability to follow a requirement [...]; the first conclusion is:

*First Conclusion*

By decreasing the effort needed to find related documents, being one of the major causes, increasing traceability is likely to increase implementation effectiveness. However, we cannot be certain about the significance of the reduction.

## 8.2 Does Improving Traceability reduce the Difficulty of Finding Relevant Information?

For testing this, six assignments have been created. The objective of every assignment is to solve it by finding the right answer to the question. These questions have been chosen in a way such that it reflects regular daily tasks. Next, for all assignments the traceability has been improved manually. In this study, the Requirements Management (RM) method that is currently used, is called M0 while the improved traceability method is called M1. Effort is measured in effectiveness (does the participant provide the right answer), efficiency (how much time does the participant need), how difficult the participant thinks the assignment was, and notes were taken if the participant had remarks or showed different behavior. The results showed that the times for solving assignments decreased when the traceability increased. Figure 8.2 shows the average number of seconds needed for finishing the different tasks. The blue line visualizes the shifting of the median; which gives an impression of the changes.

## 8.3 General Conclusion

Based on the two conclusions as stated in Section 8.1 and Section 8.2, we can answer the main research question as follows:

*Conclusion*

By improving the variables that are a major cause for low implementation effectiveness, improving traceability is likely to improve implementation effectiveness.

## 8.4 Other Conclusions

In Table 8.1, the difference between M0 Table 6.5 and M1 Table 6.6 has been calculated. The only significant difference that can be found, is the relationship between the amount of time needed to solve the assignments, and the self-rated experience in requirements management. This means, that in M1, a subject is significantly more likely to need more time for solving the assignment, if he is more experienced. This also means, that the less experienced stakeholders need less time to solve the assignments than the experienced ones. The interesting phenomenon here, is that traceability seems to remove the need for experience in solving assignments.

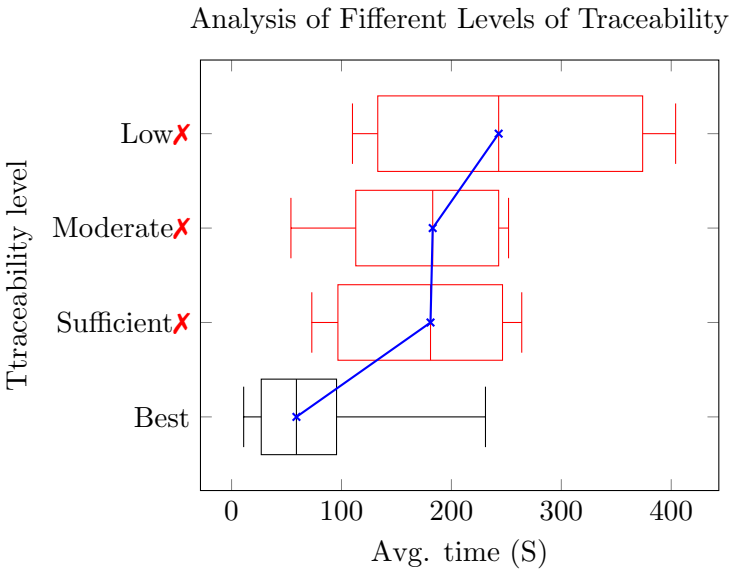



Figure 8.2: Distribution of Time Scores

TABLE 8.1: Spearman's $\rho$ Correlation Matrix of M0✓

| | T$_{(seconds)}$ | Difficulty | YearsExp | BzExpert |
|---|---|---|---|---|
| S | 1 | | | |
| Difficulty | -0.14 | 1 | | |
| YearsExp | +0.14 | +0.14 | 1 | |
| BzExpert | +0.50 | +0.08 | +0.07 | 1 |

# Chapter 9

# Future Work

This section elaborates on the future possibilities in the area of research.

As stated in Chapter 1, the topic of traceability already has a great amount of interest within the field of requirements engineering. The presence of it is considered good, some consider it the next level in requirements management [9, 24, 60]. However, this does not mean that the work is finished. First of all, to the opinion of the authors of this study, the current studies generally turn out to be too abstract to be usable. This study showed a practical appliance as a proof of concept. Second, as a large number of software product organizations may use Word and Excel too, it cannot be assumed feasible for all of them to change their working process for the purpose of adopting traceability. This study showed that with only a number of small changes and additions, full manual traceability is in fact possible. Moreover, it showed that traceability is very likely to help increasing implementation effectiveness by reducing the effort to find related data artifacts.

But this was not the only interesting result. The correlation between experience and needed time increased by .50. This implies, that good traceability provides the means for relatively new stakeholders to perform the tasks faster than the more experienced stakeholders.

## 9.1 Future Work

Since the area of Requirements Traceability (RT) is relatively young, there is a significant amount of research yet to be done. Based on this study, the authors first found out that

traceability is considered an abstract field. There is a very low number of studies that studied the practical impact of traceability, however, there is a relatively high amount of studies that analyze the topic on a more abstract level. Future research can study:

### 9.1.1 Practical Impact

This topic is mentioned multiple times in this study; this is what the authors wanted to discover. What is the practical impact for implementing requirements management? We hope, that the information that has been gathered during this study, will provide the will for future scientists to study the practical impact more. If there is reasonable interest for organizations to adopt traceability, than there may be more budget for research because of that very interest.

### 9.1.2 Framework

Another use for future research, can be the generalizability of this study. The method used in this study can be the foundation of a framework that helps organizations improving their traceability. This shall create a broader agreement on the generalizability of this study, and on the longer term, it may help proving if the method used in this study is generally generalizable.

### 9.1.3 Automation

In [61], the usage of the BugZilla and the Central Versioning System database have been combined in order to link the requirements from Bugzilla to the actual and implemented code. They concluded that automation of finding links can help in speeding op the process. Since BugZilla has a relatively big userbase, the study of [61] can be used for creating traceability links from older requirements while this study can be used for the current ones.

## 9.2 End note

The topic of RT is a very interesting one, according to the authors of this study. We hope that this research will prove to be valuable for the future, and we would like to encourage all future students and scientists to embrace this subject. Special thanks to Dr. Fabiano Dalpiaz for his support in performing this study.

# Appendix A

# Keylane—Quinity RM procedure

To read the 'Actor' column, a short description is given. C means 'Customer', while Q means 'Quinity'. Next; the ':' shows that the next actor is working for the company before the sign. The R means 'Representative', TL 'TeamLeader', Tm 'TeamMember', and Rw 'RevieWer'. Then, there are additions: .o means 'Outside the team', which means that someone from another team should do it.

The numbers show the order in which the actions take place. However, some actions contain a condition. For example, number three, if the condition is true (in this example, if the needed time is higher than three hours) then column 'True' shows what will happen.

TABLE A.1: Keylane|Quinity Requirements Management

| № | Action | Actor | Deliv. In | Deliv. | Superdeliv. | Condition | True | False |
|---|---|---|---|---|---|---|---|---|
| 1 | Describe Requirements | C:R/Q:R | RFC/ RFE/ document/ note/ bugzilla/ mail | | | | | |
| 2 | Confirm receipt of Requirements | Q:TL+:Tm | Req | Mention | Bugzilla | | | |
| 3 | Analyze Requirement + first estimation | Q:TL/:Tm.a | | LPO, FO | RFC | >3 hours | 4a | 4b |
| 4a | Create RFC[a] document | Q:TL | | RFC? | | | | |
| 4b | Create RFC; Requirements Impact; Change overview of which functions and concept will change, expire or be added | Q:Tm | | RQ Impact, changing, expiring and to be added functions and concepts | RFC | No infrastructural change. | 5 | Not this procedure |
| 5 | Document FO[b] part | Q:Tm | FO Review | | RFC | Known change/implemented? | 6a | 6b |
| 6a | Review FO | Q:TL+:Tm | FO | FO review | RFC | | | |
| 6b | Review FO | Q:TL.o+:Tm.o | FO | Review | RFC | | | |
| 7 | Wait for customer approval for FO | Q:TL | | Customer approval | RFC | Customer Approved? | 8 | 3 |
| 8 | Redo sizing FO | Q:TL | Index FO, template | Estimation | RFC | | | |
| 9 | Review.oudget estimation | ? | ? | ? | RFC | | | |
| 10 | Creating TO[c] | Q:Tm | FO | TO | RFC | Within project team? | 11a | 11b |
| 11a | Review TO | Q:TL+:Tm | TO | TO-review | RFC | | | |
| 11b | Review TO | Q:TL.o+:Tm.o | TO | TO-review | RFC | | | |
| 12 | Redo sizing with TO information | Q:TL | ? | | | | | |
| 13 | Review sizing | Q:TL+:Rw | TO | Estimation | RFC | Approved? | 14 | ? |
| 14 | Finish RFC and communicate to customer | Q:TL+:Tm | RFC | Ask for approval | | | | |
| 15 | Realization and Unittest | Q:Tm | RFC | Code+test | | | | |
| 16 | Finish realizing Inchecking Code | Q:Tm | TO | Testcode | TestQIS | | | |
| 17 | Finish realizing copying FO to functional program documentation | Q:Tm | FO | FO | RFC | | | |
| 18 | Finish realizing copying TO to technical program documentation | Q:Tm | TO | TO | RFC | | | |
| 19 | Integration test | Q:Tm | Testcode | Test report | | Approved? | 20 | ? |
| 20 | Performance and capacitytests | Q:TL+:PJM? | Testcode | Test report | | Approved? | 21 | ? |
| 21 | Check consistency between code, FO and TO(LPO) | ? | FO, LPO | Akkoord | RFC | | | |
| 22 | Review consistency check | ? | Estimation | | | Approved? | | 23 |
| 23 | Explain wrong sizing | Q:TL+:Rw | | Statement | | | | |

[a]Request For Change
[b]Functional Overview
[c]Technical Overview

# Appendix B

# Bugtrace

*[reply] [-] Private Description Barry 2015-01-16 08:56:26 CET*

+++ This bug was initially created as a clone of Bug #195282 +++

=== Achtergrond ===

companyB moet op de website persoonlijke accounts hebben, vanwege de audit trail. De accounts op de website kunnen namelijk ook aanpassingen doen in companyB.

mainProduct stuurt nieuwe expertiseopdrachten via een webservice is, deze heeft een eigen, ander account. Deze inlogfunctionaliteit van beide is gescheiden.

mainProduct toont op de detailschermen in mainProduct links naar de site van companyB om nog extra details te bekijken (mainProduct haalt deze niet op bij companyB). mainProduct zorgt er nu voor dat er een username + password worden gegeven, en met RFC UV–S–XX3 regelen we onder water een sessiecode die we meegegeven, zodat de usernames en passwords niet in de schermen komen te staan.

=== Probleem ===

Bij het raadplegen via Schade, Communicatie, Expertiseopdrachten is een externe link naar companyB om de schade in te kunnen zien, bijvoorbeeld: LINK

Wanneer nu twee verschillende gebruikers een dossier willen raadplegen raakt de 1e zijn sessie kwijt.

De volgende tekst komt uit de FAQ van companyB:

Sessies met de centrale companyB servers verlopen ook direct wanneer twee personen tegelijk met dezelfde key in het systeem inloggen. Wilt u met meerdere personen tegelijk in companyB kunnen werken, dan heeft u daarvoor extra keys nodig. Extra keys kunt u aanvragen via uw expert of door een mailtje te sturen naar info@companyB.nl

=== Wens ===

mainProduct moet voor toegang tot de _website_ persoonlijke accountnamen doorgeven, zodat in companyB uiteindelijk te achterhalen is wie er ingelogd is geweest en wat ze precies hebben gedaan. Dus een user in mainProduct moet een eigen user op de website van companyB hebben.

Omdat companyB voor meerdere maatschappijen werkt, kunnen ze niet garanderen dat loginnamen uit mainProduct uniek zijn in het systeem, dus die doorgeven is niet handig. Ze geven nu ook aan dat logins beperkt zijn op 10 tekens; maar dat zie ik meer als technische uitdaging voor ‘hen’.

=== Oplossingsrichting ===

De oplossingsrichting die customer en companyB nu voor ogen hebben, is om bij een gebruiker in mainProduct de gebruikersnaam-voor-companyB vast te leggen, en die voor de gebruikers die op het linkje op het scherm klikken door te geven (ofwel met XX3 te gebruiken om de sessie te genereren). Met de username geven we dan een customer-breed wachtwoord mee waardoor je geen wachtwoordbeheer per gebruiker hebt, maar m maar 1 x hoeft in te stellen.

==> Op basis hiervan vul ik de milestone om ’m alvast op de planning en in het vizier te krijgen. Ik zal de details nog even met Anouk afstemmen.

*[reply] [-] Private Comment 1 Pieter 2015-01-16 10:54:59 CET*

Barry, Gwennie en ik bespreken dit vanmorgen, daarna wordt deze bug bijgewerkt met de gekozen oplossingsrichting.

*[reply] [-] Private Comment 2 Gwennie 2015-01-16 12:43:44 CET*

Created attachment 167668 [details]

Mailconversatie met mogelijke oplossingen

STATUS:

Besproken met Barry en Pieter.

OPLOSSING

Zie ook attachment.

```
45 Het idee is om externe gebruikersnamen vast te leggen in mainProduct bij een
       gebruiker (logonUser). Het is nodig om meerdere externe
     gebruikersnamen per gebruiker vast te leggen, omdat een gebruiker
     aparte gebruikersnamen heeft voor companyBproductA en companyBproductB.
     Voor companyB zijn dit eigenlijk twee verschillende systemen.
46 Deze externe gebruikersnamen kunnen we dan opslaan in een aparte tabel. In
     deze nieuwe tabel kunnen we dan ook de operator-id opnemen.
47
48 VERVOLG:
49 We moeten hiervoor op korte termijn een RFC opstellen. Als deze snel
     akkoord is kunnen we het testen combineren met UV-S-XX3.
50
51
52 [reply] [-] Private Comment 3 Jeanine van Mersbergen 2015-01-26 08:17:28 CET
53 LINK
54 Datum: 23-01-2015
55 Aangeboden door: Jeanine van Mersbergen
56 Reviewer(s): Annemieke en Jeroen
57
58 Wat is gereviewd:
59 RFC-UV-S196595 Voorkomen verlopen sessie companyB v0.0.1
60
61 Vervolgafspraken & opmerkingen:
62 Het doel van de RFC moet duidelijker naar voren komen.
63 We voegen een nieuwe tabel toe 'ExternalUserType'. We zetten een
     koppeltabel tussen LogonUser en de nieuwe tabel.
64 Navragen bij companyB of zij iets met het wachtwoord doen dat mainProduct
     meestuurt.
65 Ik ga dit verwerken in een nieuwe versie.
66
67
68 [reply] [-] Private Comment 4 Jeanine 2015-01-26 11:39:35 CET
69 In comment #2 staat beschreven dat we operatorId ook verplaatsen naar de
     nieuwe tabellen. Afgestemd met Jeroen en Annemieke om dit niet te doen,
      omdat dit veld functioneel anders gebruikt wordt dan de nieuwe velden
     bij gebruiker.
70
71
72 [reply] [-] Private Comment 5 Pieter 2015-01-28 15:52:18 CET
73 Ik heb een sizing aangemaakt op LINK
74
75
76 [reply] [-] Private Comment 6 Bob 2015-02-06 14:02:49 CET
77 Review: Sizing
78 Datum: 06-02-2015
79 Aangeboden door: Pieter
80 Reviewer(s): Bob
81 Vervolgreview nodig: Nee
```

```
82 Wat is gereviewd:
83 LINK
84
85 Vervolgafspraken & opmerkingen:
86 Volgende punten toegevoegd aan RFC en review:
87 Autorisatie op tabblad 'Externe gebruikersgegevens'.
88 Bijhouden historie voor wijzigingen op 'Externe gebruikersgegevens'.
89
90 Wijzigingen zijn verwerkt in v0.2 van de sizing.
91
92
93 [reply] [-] Private Comment 7 Bob 2015-02-06 14:03:55 CET
94 RFC en sizing staan hier:
95 LINK
96
97 Barry, kun jij deze bug weer bij mainProduct Schade zetten als het akkoord
       is om de RFC op te pakken?
98
99
100 [reply] [-] Private Comment 8 Barry 2015-02-10 12:04:14 CET
101 STATUS: Wacht op akkoord.
102
103
104 [reply] [-] Private Comment 9 Ellen 2015-05-27 11:55:19 CEST
105 Ter info: Het customer RFC nummer is 681.
106
107
108 [reply] [-] Private Comment 10 Barry 2015-07-14 09:18:06 CEST
109 Akkoord ontvangen van customer.
110
111 @Pieter: We moeten even bellen wanneer deze past.
112
113
114 [reply] [-] Private Comment 11 Pieter 2015-07-14 09:46:11 CEST
115 Marjan, zou je deze in de FO's willen verwerken?
116
117
118 [reply] [-] Private Comment 12 Marjan 2015-07-14 16:20:25 CEST
119 Ik heb de RFC verwerkt in het datamodel.
120
121 Verwijderde velden:
122 - Businessunit.companyBHouseWebsiteUsername
123 - Businessunit.companyBHouseWebsitePassword
124 - Businessunit.companyBCarWebsiteUsername
125 - Businessunit.companyBCarWebsitePassword
126
127 Toegevoegde tabellen:
128 - ExternalUserType
```

```
129 - LogonUserExternalUserType
130
131 Ik heb de RFC verwerkt in de FO's.
132
133 LINK
134 3.2.2 Beheren externe gebruikersgegevens
135 4.2.3 Tonen historieregels in lijst
136
137 LINK
138 4.3.1 Opvragen sessienummer
139
140 LINK
141 4.1.2 Raadplegen expertiseopdracht
142
143 LINK
144 3.2 Toevoegen / wijzigen business unit
145
146 LINK
147 4 Historie bij externe gebruikersgegevens
```

Listing B.1: "Bugtrace in Bugzilla"

# Bibliography


[1] Evert Duipmans and Dr L Ferreira Pires. Business process management in the cloud: business process as a service (bpaas). *University of Twente*, 2012. pages xvii, 15

[2] Motoshi Saeki. Embedding metrics into information systems development methods: An application of method engineering technique. In *Advanced information systems engineering*, pages 374–389. Springer, 2003. pages xvii

[3] Inge van de Weerd, Johan Versendaal, and Sjaak Brinkkemper. A product software knowledge infrastructure for situational capability maturation: Vision and case studies in product management. *Technical Report UU-CS*, 2006:1–16, 2006. pages xvii

[4] Inge van de Weerd, Sjaak Brinkkemper, and Johan Versendaal. Concepts for incremental method evolution: empirical exploration and validation in requirements management. In *Advanced information systems engineering*, pages 469–484. Springer, 2007. pages xvii

[5] Inge van de Weerd, Sjaak Brinkkemper, Richard Nieuwenhuis, Johan Versendaal, and Lex Bijlsma. Towards a reference framework for software product management. In *Requirements Engineering, 14th IEEE International Conference*, pages 319–322. IEEE, 2006. pages xvii

[6] Orlena CZ Gotel and Anthony CW Finkelstein. An analysis of the requirements traceability problem. In *Requirements Engineering, 1994., Proceedings of the First International Conference on*, pages 94–101. IEEE, 1994. pages xviii, 1, 13, 14, 18, 24, 25, 26, 44, 77, 88

[7] Centers for Medicare & Medicaid Services. Choosing an appropriate system development methodology. `https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf`, February 2015. (Accessed on 07/22/2016). pages xviii, 32

[8] Orlena Gotel, Jane Cleland-Huang, J Huffman Hayes, Andrea Zisman, Alexander Egyed, Paul Grünbacher, and Giuliano Antoniol. The quest for ubiquity: A roadmap for software and systems traceability research. In *Requirements Engineering Conference (RE), 2012 20th IEEE International*, pages 71–80. IEEE, 2012. pages 1, 2

[9] Jim Heumann. The five levels of requirements management maturity. *the Rational EDGE*, 2003. pages 1, 2, 15, 20, 22, 23, 25, 91

[10] Alan Hevner and Samir Chatterjee. *Design science research in information systems*. Springer, 2010. pages 5

[11] Kathleen M Eisenhardt. Building theories from case study research. *Academy of management review*, 14(4):532–550, 1989. pages 5

[12] Kenneth A Frank. Impact of a confounding variable on a regression coefficient. *Sociological Methods & Research*, 29(2):147–194, 2000. pages 6

[13] Balasubramaniam Ramesh. Factors influencing requirements traceability practice. *Communications of the ACM*, 41(12):37–44, 1998. pages 7, 8, 27, 28, 29

[14] Robert Weissberg and Suzanne Buker. *Writing up research*. Prentice Hall Englewood Cliffs, NJ, 1990. pages 13

[15] Claes Wohlin. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, page 38. ACM, 2014. pages 13

[16] Samireh Jalali and Claes Wohlin. Systematic literature studies: database searches vs. backward snowballing. In *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement*, pages 29–38. ACM, 2012. pages 13

[17] Michael Edwards and Steven L Howell. A methodology for systems requirements specification and traceability for large real time complex systems. Technical report, DTIC Document, 1991. pages 13

[18] VL Hamilton and ML Beeby. Issues of traceability in integrating tools. In *Tools and Techniques for Maintaining Traceability During Design, IEE Colloquium on*, pages 4–1. IET, 1991. pages 14, 17

[19] Balasubramaniam Ramesh and Matthias Jarke. Toward reference models for requirements traceability. *Software Engineering, IEEE Transactions on*, 27(1):58–93, 2001. pages 14

[20] Martin Shepperd and Gada Kadoda. Comparing software prediction techniques using simulation. *Software Engineering, IEEE Transactions on*, 27(11):1014–1022, 2001. pages 15

[21] Bo Xiao and Izak Benbasat. E-commerce product recommendation agents: Use, characteristics, and impact. *Mis Quarterly*, 31(1):137–209, 2007. pages 15

[22] Francisco AC Pinheiro and Joseph A Goguen. An object-oriented tool for tracing requirements. In *Requirements Engineering, 1996., Proceedings of the Second International Conference on*, page 219. IEEE, 1996. pages 15

[23] Robin F Goldsmith. *Discovering real business requirements for software project success*. Artech House, 2004. pages 15, 19, 21, 44

[24] Tyler Blain. Cmmi levels and requirements management maturity introduction — tyner blain. `http://tynerblain.com/blog/2007/01/25/cmmi-and-rmm-intro/`, 01 2007. (Visited on 07/14/2015). pages 15, 22, 25, 91

[25] Peter Sawyer, Ian Sommerville, and Stephen Viller. Requirements process improvement through the phased introduction of good practice. *Software Process: Improvement and Practice*, 3(1):19–34, 1997. pages 15

[26] Angelina Espinoza and Juan Garbajosa. A study to support agile methods more effectively through traceability. *Innovations in Systems and Software Engineering*, 7(1):53–69, 2011. pages 15

[27] Herb Sutter. The free lunch is over: A fundamental turn toward concurrency in software. *Dr. Dobb's journal*, 30(3):202–210, 2005. pages 16

[28] Dean Leffingwell and Don Widrig. *Managing software requirements: a unified approach*. Addison-Wesley Professional, 2000. pages 16, 19, 25, 26

[29] Joseph Fiksel and Mark Dunkle. Principles of requirement management automation. In *Reliability and Maintainability Computer-Aided Engineering in Concurrent Engineering, 1990 and 1991., Combined Proceedings of the 1990 and 1991 Leesburg Workshops on*, pages 231–236. IEEE, 1992. pages 16

[30] Stefan Winkler and Jens Pilgrim. A survey of traceability in requirements engineering and model-driven development. *Software and Systems Modeling (SoSyM)*, 9(4):529–565, 2010. pages 17

[31] Sol J Greenspan and Clement L McGowan. Structuring software development for reliability. *Microelectronics Reliability*, 17(1):75–83, 1978. pages 17

[32] Tim Smithers, Ming Xi Tang, and Nils Tomes. The maintenance of design history in ai-based design. 1991. pages 17

[33] William H Roetzheim. *Developing software to government standards*. Prentice-Hall, Inc., 1991. pages 17

[34] Wei Ding, Peng Liang, Antony Tang, and Hans Van Vliet. Knowledge-based approaches in software documentation: A systematic literature review. *Information and Software Technology*, 56(6):545–567, 2014. pages 17

[35] Karl Wiegers. Requirements traceability: Links in the requirements chain, part 1, 2013. URL `http://www.jamasoftware.com/blog/requirements-traceability-links-in-the-requirements-chain-part-1/`. pages 17, 18, 19, 39

[36] Alan M Davis. *Software requirements: objects, functions, and states*. Prentice-Hall, Inc., 1993. pages 17, 19

[37] Nawazish Ali, Yann-Gael Gueneuc, and Giuliano Antoniol. Trustrace: Mining software repositories to improve the accuracy of requirement traceability links. *Software Engineering, IEEE Transactions on*, 39(5):725–741, 2013. pages 17

[38] Orlena Gotel, Jane Cleland-Huang, Jane Huffman Hayes, Andrea Zisman, Alexander Egyed, Paul Grünbacher, Alex Dekhtyar, Giuliano Antoniol, Jonathan Maletic, and Patrick Mäder. Traceability fundamentals. In *Software and Systems Traceability*, pages 3–22. Springer, 2012. pages 17

[39] Muhamamd Farhan Bashir and Muhammad Abdul Qadir. Traceability techniques: A critical study. In *Multitopic Conference, 2006. INMIC'06. IEEE*, pages 265–268. IEEE, 2006. pages 17

[40] Michael G Christel and Kyo C Kang. Issues in requirements elicitation. Technical report, DTIC Document, 1992. pages 18

[41] Antony Tang, Yan Jin, and Jun Han. A rationale-based architecture model for design traceability and reasoning. *Journal of Systems and Software*, 80(6):918–934, 2007. pages 19

[42] Keith Ellis. Business analysis benchmark: The impact of business requirements on the success of technology projects, 2015. URL `http://www.iag.biz/images/resources/iagbusinessanalysisbenchmark-fullreport.pdf`. pages 19

[43] Jane Radatz, Anne Geraci, and Freny Katki. Ieee standard glossary of software engineering terminology. *IEEE Std*, 610121990(121990):3, 1990. pages 20

[44] Willem Bekkers and I van de Weerd. Spm maturity matrix. *Utrecht University*, 2010. pages 22

[45] Scott Sehlhorst. What cmmi level should we use? — tyner blain, 2006. URL `http://tynerblain.com/blog/2006/03/12/what-cmmi-level-should-we-use/`. pages 22

[46] Sergio España Cubillo. *Methodological integration of Communication Analysis into a Model-Driven software development framework*. PhD thesis, Universitat Politècnica de València, 2011. pages 24, 25

[47] Alan Davis, Oscar Dieste, Ann Hickey, Natalia Juristo, and Ana M Moreno. Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review. In *Requirements Engineering, 14th IEEE International Conference*, pages 179–188. IEEE, 2006. pages 24, 25

[48] Claire Ingram and Steve Riddle. Cost-benefits of traceability. In *Software and Systems Traceability*, pages 23–42. Springer, 2012. pages 25, 51

[49] Jane Cleland-Huang, Grant Zemont, and Wiktor Lukasik. A heterogeneous solution for improving the return on investment of requirements traceability. In *Requirements Engineering Conference, 2004. Proceedings. 12th IEEE International*, pages 230–239. IEEE, 2004. pages 25, 51

[50] A Terry Bahill and Steven J Henderson. Requirements development, verification, and validation exhibited in famous failures. *Systems Engineering*, 8(1):1–14, 2005. pages 25

[51] Paul Rook. Controlling software projects. *Software Engineering Journal*, 1(1):7, 1986. pages 31, 32

[52] S Balaji and M Sundararajan Murugaiyan. Waterfall vs. v-model vs. agile: A comparative study on sdlc. *International Journal of Information Technology and Business Management*, 2(1):26–30, 2012. pages 32

[53] Robyn Longhurst. Semi-structured interviews and focus groups. *Key methods in geography*, pages 117–132, 2003. pages 40

[54] Yvonne M Williamson. *Research methodology and its application to nursing*. John Wiley & Sons, 1981. pages 40

[55] K Louise Barriball and Alison While. Collecting data using a semi-structured interview: a discussion paper. *Journal of advanced nursing*, 19(2):328–335, 1994. pages 40

[56] Hassan Gomaa and Douglas BH Scott. Prototyping as a tool in the specification of user requirements. In *Proceedings of the 5th international conference on Software engineering*, pages 333–342. IEEE Press, 1981. pages 47

[57] Jeane W Anastas. *Research design for social work and the human services*. Columbia University Press, 1999. pages 57

[58] Samuel Sanford Shapiro and Martin B Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, 1965. pages 73

[59] James D Evans. *Straightforward statistics for the behavioral sciences*. Brooks/Cole, 1996. pages 75, 76

[60] Rong Peng, Qiang Ye, and Mao Ye. A requirements maturity measurement approach based on sklsewiki. In *Computer Software and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual*, pages 251–254. IEEE, 2010. pages 91

[61] Kamel Ayari, Peyman Meshkinfam, Giuliano Antoniol, and Massimiliano Di Penta. Threats on building models from cvs and bugzilla repositories: the mozilla case study. In *Proceedings of the 2007 conference of the center for advanced studies on Collaborative research*, pages 215–228. IBM Corp., 2007. pages 92