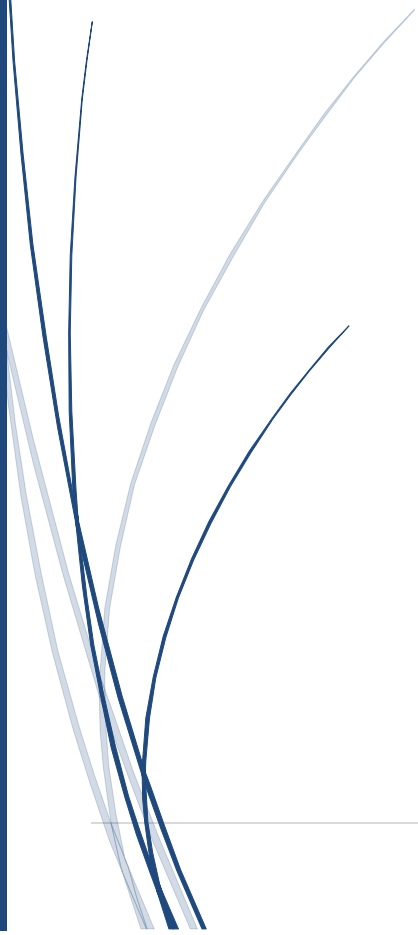
Software Traceability

Investigating the perceived value of software traceability in practice

Jin Yang Hu

SUPERVISORS:
MARCELA RUIZ & FABIANO DALPIAZ

12/11/2019



Utrecht University

Utrecht University
Department of Information and Computing Sciences

Master Thesis Business Informatics

Investigating the perceived value of software traceability in practice

Jin Yang Hu

j.y.hu@students.uu.nl

Supervisors
dr. L.M. Ruiz Carmona
dr. F. Dalpiaz

December 2019

Abstract

Introduction. Many advances have been made in the research regarding software traceability, regarding the tools, methods, and techniques to create, maintain and use traceability. Nonetheless, even with the well understood importance of traceability in the scientific software engineering community, according to researchers, traceability is still, “a sought-after, yet often elusive quality in software-intensive systems”, and the realization, awareness, and the need for software traceability varies from context to context and on a project by project basis. Therefore, multiple researchers indicated that there is still much need for empirical knowledge regarding the use of traceability within the typical domains, which include traditional software development methods and safety critical projects, but also outside these typical domains.

Method. In this study we present the empirical knowledge gained regarding the perceived value, in terms of the current situation and the needs, of practitioners that use software traceability in practice. First, we investigate common perceptions and problems encountered in practice according to the current literature. Second, we deploy an online survey-questionnaire to gain the perceptions of practitioners on these problems and their needs. Third, we conduct multiple semi-structured interviews with participants of the online survey-questionnaire to gain more knowledge regarding their answers on the survey. Furthermore, the participants and the results of the questionnaire are mainly evaluated and compared based on the different development paradigms in terms of Agile, Traditional, or Mixed software development life cycles, and on the type of project in terms of safety critical and non-safety critical. The evaluation is done based on the overall sentiment analysis of the participants. In addition, a Wilcoxon analysis is performed to see if there are any significant differences.

Results. 55 practitioners participated in the online survey-questionnaire. The quantitative Wilcoxon analysis showed that there is one significant difference in the response distribution between the Agile and Traditional groups, in which the participants in the Traditional group agreed more strongly that traceability is mostly performed manually than the Agile group with an effect size margin of 0.94, and a p-value of 0.048 ($p < 0.05$). The sentiment analysis showed based on the collective results of all the participants that regardless of which development paradigm was followed and project type in terms of safety-critical or not, that similar perceptions and traceability situations are perceived, such as that traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices, that traceability is of high importance for the software development process, and that traceability is mostly performed manually. In addition, despite small differences in the rankings, similar needs are perceived as high priority as well, such as that there is a need to perform traceability in a more managed fashion, that there is a need for a more clear overview of the costs and benefits regarding traceability, that there is a need for more traceability automation, that there is a need to increase the awareness of the importance of traceability, and that there is a need for more collaboration and guidance regarding traceability.

Discussion. These results suggest that the paradigm and the type of project in terms of safety critical or not, do not significantly affect the software traceability situations and needs of practitioners, which indicates that the perceptions, and problems, and needs, found can be generalized to a certain extend from those perspectives. However, even though the paradigm and project type might not significantly influence the perceptions of practitioners, they might play a more subtle role. In addition, factors, such as the tools used, the tool setup, company size and structure, and more have to be taken into account and potentially play a bigger role in the differences found between traceability usage, perceptions, problems, and needs, found in practice.

Keywords. Software traceability _ Perceived value _ Needs _ Software traceability situation _ Agile _ Safety critical _ Online survey-questionnaire _ Empirical data

Acknowledgements

In December 2018, I started on a journey to investigate the value of software traceability. The initial idea was to design a kind of value framework that could be used by practitioners in practice. However, the idea was still very rough, and I had little knowledge regarding the research and practice of software traceability. Over the months, I immersed myself in the traceability studies that have been done by the traceability research communities and learned that much work has been done, but that there was still a need for more studies in this field, especially regarding empirical knowledge. After multiple sessions with my supervisors, we eventually decided to change the initial idea and to focus on a more empirical approach where we would try to gain more insight in the current practice and needs of practitioners regarding traceability to support researchers and practitioners alike.

By having conducted this study, I have learned of the difficulty of conducting these kinds of empirical studies in which I was challenged to be a critical thinker and a specialist to understand the field of software traceability and to be able to come up with and discuss ideas with the supervisors, to be a good explainer and a generalist as to be able to bridge the gap between the terminology used between researchers and practitioners, to improve socially to be able to advertise the study in search of participants, to improve my interview skills to be able to have meaningful conversations with the participants, and also to sharpen my coding and scripting skills to be able to analyze the gathered data in R. Furthermore, by having conducted a study in the area of software traceability, I have learned how challenging it is to conduct a study in this field but also learned about the opportunities present in the research area of software traceability. I hope that the insights obtained in this study and the data gathered can contribute to the needs of the researchers and in turn the practitioners as well.

This study would not have been possible without my supervisors. Firstly, I would like to thank Marcela Ruiz for her guidance and positivity. Your ambitious mindset and enthusiasm gave me the extra motivation to conduct the study and your knowledge and valuable insights regarding software traceability greatly helped in shaping the study. Secondly, I would like to thank Fabiano Dalpiaz for his guidance and positivity as well. The fact that you were involved early during the study and were just as invested was very valuable. Your ambitious but also more grounded mindset helped in balancing and scoping the study, and your knowledge and insights based on your expertise of requirements traceability greatly helped in the progression of the study. In addition, I really enjoyed the meetings we had with the three of us and the insightful conversations. Third, I would like to thank the participants of the validation in the shaping and refinement of the survey. Fourth, I thank all the practitioners that helped in the distribution of the survey and the filling in of the survey. Furthermore, a special thanks for the practitioners that were willing and provided their time for an interview.

Finally, I would also like to thank my family and friends for their support and encouragements to keep me going. Without any of the people above, this study would not have been possible.

Jin Yang (Jimmy) Hu

Contents

1. Introduction

Software and requirements traceability are an important aspect of software engineering and the importance is well understood by the software engineering community. According to the Center of Excellence for Software and Systems Traceability (CoEST)[1], which is an open community for both academics and industry, requirements traceability is 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 ongoing refinement and iteration in any of these phases*"

Software traceability extends the definition to encompass multi-directional traceability centered around diverse artifacts, and as such is:

> "*the ability to be able to interrelate uniquely identifiable software engineering artifacts to each other to maintain required links over time and use the resulting network to answer questions of both the software product and its development process*"

The possible uses of traceability are, but not limited to, supporting the verification, testing, and implementation of requirements, satisfying regulatory checks of the government, analyzing the impact of changes on the current software system and its requirements, safety analysis, project intelligence and possibly much more.

Nonetheless, even with the well understood importance of traceability in the scientific software engineering community, according to Cleland-Huang et al., (2014), traceability is still, "a sought-after, yet often elusive quality in software-intensive systems", and the realization, awareness, and the need for software traceability varies from context to context and on a project by project basis.

1.1 Motivation & problem statement

In the academia, traceability researchers have set up several challenges to ultimately make traceability as effortless as possible for software engineers in practice so that they can focus their time and energy on other important aspects in the software engineering life cycle. In addition, it ultimately accelerates and increases the quality and efficiency of software engineering and the developed systems and software (Orlena Gotel et al., 2012). This in turn might increase customer satisfaction, which is becoming of utmost importance for businesses that also see an increasing adoption of Agile software development life cycles (SDLC) and DevOps in the development (COLLAB.NET & VERSIONONE.COM, 2018).

The biggest challenge, also referred to as the grand challenge of traceability, is to make traceability ubiquitous (Orlena Gotel et al., 2012). This means that it will be purely in the background, trusted by all stakeholders, and that it supports all the phases in the software and systems engineering life cycles. To be able to tackle the grand challenge of traceability, more research and progress must be made regarding the surrounding challenges that ultimately lead to ubiquitous traceability. These include challenges surrounding the purpose, cost-effectiveness, configurability, trust, scalability, portability, and the value of traceability. As such, the researchers try to raise the awareness of traceability and the importance on doing more research on it in the context of software engineering. They describe that traceability is an important and critical ability in software intensive systems and that by actively managing traceability it could bring several to many benefits. Their ideal vision is that in 2035,

[1] http://www.coest.org/

traceability is ubiquitous, active in the background, and all surrounding. Even though there have been advances regarding the research on traceability in the past decade, much more research is needed to be able to make traceability ubiquitous in 2035.

Currently, much of the traceability research indicate to be more focused on the techniques, methods, and tools to generate and automate trace links between requirements and code and the related software artifacts (Antoniol, Cleland-Huang, Hayes, & Vierhauser, 2017; Cleland-Huang et al., 2014; O. Gotel et al., 2012; Orlena Gotel et al., 2012). One of the primary reasons is that manual traceability is usually reported as slow, requiring much effort and that the artefacts and trace links produced in this way are usually quickly out of date (Ingram & Riddle, 2013). However, compared to the research on techniques, methods, and tools to automate trace links, there is less transparency on the current state of practice and actual usage of software traceability in the industry by practitioners and the related perceived expected benefits, costs, and challenges (Gotel et al., 2012; Cleland-Huang et al., 2014). As such, there is still a need to increase the body of knowledge surrounding the usage of software traceability in the software industry and the benefits, limitations, and needs, that practitioners face when implementing and maintaining traceability.

In practice, according to Cleland-Huang et al. (2014), traceability gets more attention in regulated and/or safety critical domains, while in other areas there is less need or where practitioners are not even aware of the term software traceability. In practice, traceability is often conducted in an ad-hoc, after-the-fact manner because of a variety of different reasons, including lack of tool support and supporting methods. Therefore, its benefits are not always fully realized. Engineers must come up with ad-hoc traceability methods to exploit trace links in practice and current methods and tools do not fully support the evolving stakeholders' needs when making use of trace links in different contexts (Gotel et al., 2012).

As such, there is currently a perceived:

- Imbalance in the empirical knowledge of traceability in practice between critical safety and non-critical safety projects
- Imbalance in the empirical knowledge of traceability in practice between Agile and Traditional development traceability research based on the increasing adoption of Agile development methods
- Imbalance between theoretical and empirical research and a need for more contemporary empirical knowledge about software traceability in practice regarding stakeholder perceptions, their current practice, challenges, and their needs, on software traceability,

These problems serve as the main driver and scope for this study and help us formulate our research objective which is described in the following section. The described problems and potential gaps of traceability regarding research and practice are described in more detail in the literature study, section 3.

1.2 Research objectives & contributions

Based on the problems and needs described in the previous section, the main goal of this research project is to investigate the differences in the perceived value in terms of practitioner's current software traceability situation and their needs of software traceability in practice. The research objective of this study is formulated as follows:

> *To explore the perceived value of software traceability in practice and its differences between Agile and non-Agile software development life cycles and safety critical and non-safety critical projects in*

> *order to gain more contemporary insight in the current software traceability situation and needs in practice.*

By conducting this study, we see several contributions:

- Gain more insight regarding software traceability in practice between the typical domains and non-typical domains.
- Add to the empirical body of knowledge of software traceability.
- Serves to update and validate previous findings.
- Collecting contemporary evidence to possibly position future research.

1.3 Why perceived value?

When evaluating to purchase or not to purchase a product or service, one of the important and main underlying drivers is most likely the perceived value in which you make a tradeoff between the expected benefits compared to the expected costs that you perceive. Even though software traceability is not a product or service per se, we do think that the perceived value of software traceability plays an important role in describing the variance in the adoption and effort made to establish and manage traceability in practice, especially in domains where software traceability is not mandated by government regulations. Furthermore, it is important to note that perceived value can not only be viewed from the cost and benefit perspective but can be viewed in a variety of different ways which is further explained in section 3.8.

In this study, we follow the means-end perspective and follow the definition of perceived value according to the Cambridge dictionary:

> "*perceived value is the value of a product based on how much customers want or need it, rather than on its real price*"

When we put this definition in the context of Software traceability, perceived value can be an indication of how much software traceability is wanted or needed in a specific context. By investigating the perceived value in this context, we will be able to touch upon the needs of practitioners and their current software traceability situation and explore, and investigate some of the challenges that are stated in the document of the grand challenges of traceability, such as the cost-benefit problem, different stakeholder views, and other challenges (Orlena Gotel et al., 2012), which are described in further detail in section 3.

1.4 Research questions

To address the main objective, this research is structured around the following main research question (MRQ):

MRQ	*"How is the perceived value of software traceability affected between Agile and Traditional development life cycles and between safety-critical and non-safety critical projects?".*

To help answer this main research question, we have formulated the following research and sub research questions:

RQ1	"*What is the current state of software traceability in research and practice*?"
RQ1.1	"*What is the current state of research regarding software traceability?*"
RQ1.2	"*What is the current state in practice regarding software traceability?*"

RQ2	"*What is the reported value of software traceability in the software development life cycle?"*
RQ2.1	"*What are the reported benefits of software traceability in the literature?*"
RQ2.2	"*What are the reported costs of software traceability in the literature?"*

The first two research questions aim to establish the theoretical background of software traceability to be able to conduct the study. Based on the results of these questions the main research question was refined and further research questions were specified. The research approach is described in section 2. Based on the literature review the following more specific research questions were formulated and refined:

RQ3	*"How can the perceived value of practitioners regarding software traceability be measured?"*

RQ4	*"Is the perceived value of software traceability affected by the type of software development life cycle?"*
RQ4.1	*"Is the current software traceability situation affected by the type of software development life cycle?"*
RQ4.2	*"Are the software traceability needs affected by the type of software development life cycle?"*

RQ5	*"Is the perceived value of software traceability affected by the type of software project?"*
RQ5.1	*"Is the current software traceability situation affected by the type of software project?"*
RQ5.2	*"Are the software traceability needs affected by the type of software project?"*

Research questions 3, is answered by conducting a literature study related work that involves perceived value. Research questions 4 and 5 are answered by executing the empirical part of this study which is described in the following section. By answering these research questions, the goal is to answer the main research question partially from a theoretical perspective based on the literature but mainly from an empirical perspective.

2. Research design

2.1 Design science

This study follows the empirical cycle of Wieringa, (2014). Initially, this research project followed the design science methodology to answer the initial main research question. Design science is the design and study of artefacts in a specific context, in which the goal is to improve or solve a problem. It consists of four main phases in an iterative cycle as show in figure 1, the problem investigation, treatment design, treatment validation, and treatment implementation. For this study, the initial focus was to design an artifact according to the previously mentioned cycle, but it was not yet clear what and how the artifact would look like. Therefore, a part of this research design is also situated in grounded theory, in which studies are designed and refined based on gathered data, which could for example be literature and interviews.

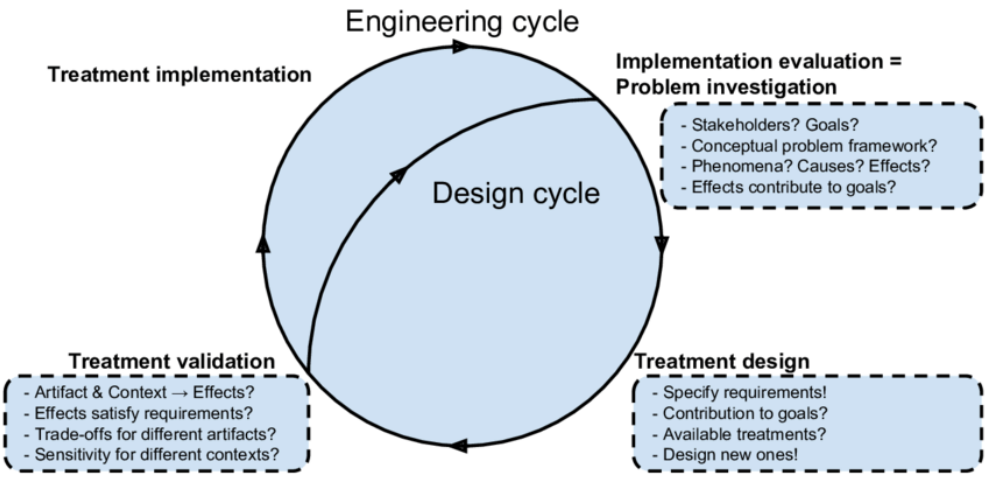


Figure 1: Design Science Engineering cycle of Wieringa, (2014).

However, during the problem investigation it became clear that there is a need for more contemporary empirical data from the perspective of practitioners regarding software traceability. As such, instead of continuing with the treatment design and validation phases from the design cycle, the remainder of the study after the problem investigation follows the empirical cycle to answer several knowledge questions specified based on the problem investigation. The empirical cycle as shown in figure 2 is also part of the design science approach of Wieringa (Wieringa, 2014).

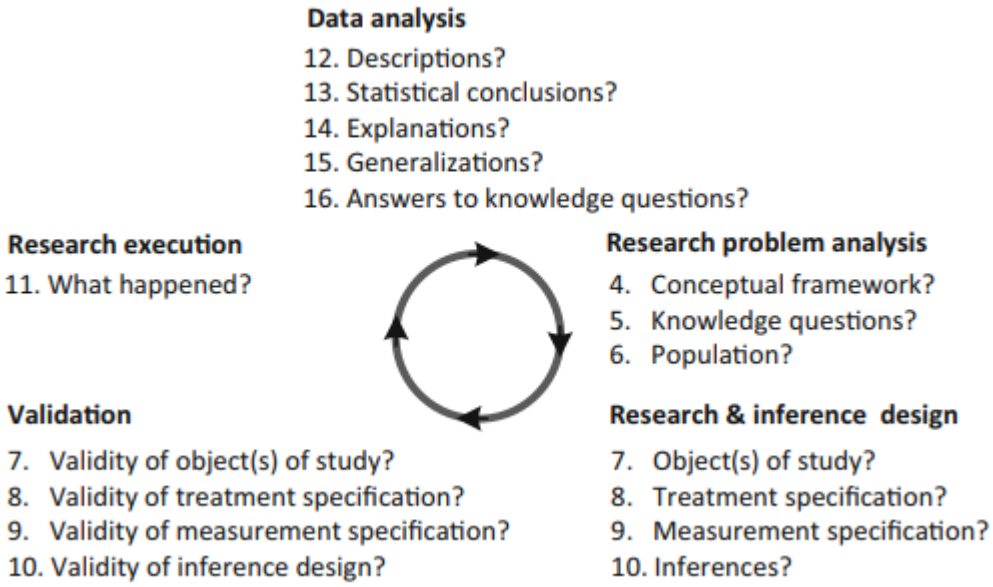


Figure 2: Empirical design cycle (Wieringa, 2014).

2.1.1 Problem investigation & Research problem analysis

The main goal of the problem investigation, which is similar to the research and problem analysis phase, is to get a more detailed and better understanding of the context in which the research is aimed towards and executed (Wieringa, 2014).

Therefore, a literature review with a mixed database search and snowballing approach is conducted based on recent literature to answer RQ1, RQ2 and RQ3 to:

- Understand the fundamentals of software traceability to conduct this study
- Understand the current state of research regarding software traceability
- Understand and or find current research challenges in the software traceability community
- Understand the current state of practice of software traceability in more detail
- Understand the differences of software traceability between agile and non-agile environments
- To gather and investigate relevant literature regarding the value, benefits, costs, challenges and use cases of Software traceability
- To investigate what perceived value is and the perspectives it can be viewed from.

Based on the literature review, the initial set of research questions are refined, and more specific questions were specified, which structures the rest of the study and serves as input for the research & inference design.

As result of the literature study, the conceptual research framework as shown in figure 3**,** was designed to give a better overview of the concepts, objects, and focus of this study.

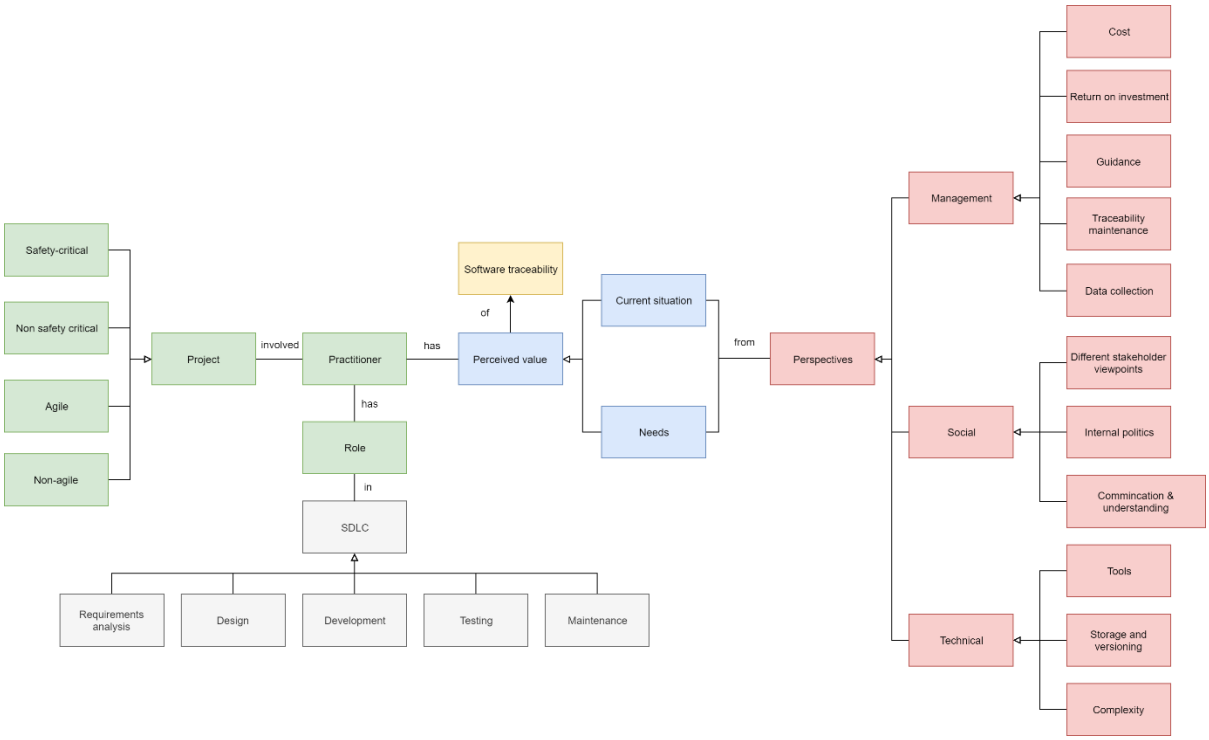


Figure 3: Conceptual research framework

As can be seen in the framework, this study investigates the perceived value of practitioners regarding software traceability in terms of current situation and needs, as described in section 1.3 and in further detail in section 3.8.

The main distinctions that are investigated is between practitioners in safety-critical and non-safety critical projects and between practitioners in Agile and Traditional development environments. Beside these main distinctions, other contextual and demographic factors such as software type and practitioner roles are investigated as well.

2.1.2 Research & inference design

In this phase, decisions are made on the objects of study, treatment specification, measurement specification, and inferences are made to how the research setup answers the knowledge questions.

To answer research RQ3, related literature about perceived value is investigated to understand how perceived value can be measured. The results of this review together with the results of the literature review of RQ1 and RQ2, serves as input for the design of the questionnaire and its contents.

As previously described, the design of the questionnaire is based on the results of the literature review. In addition the contents of the questionnaire, in particular the questions, are created based on the GQM approach (Basili, Caldiera, & Rombach, 1994; van Solingen Rini, Vic, Gianluigi, & Dieter, 2002). The approach and design of the content of the questionnaire is described in more detail in section 4.

The questionnaire is used to gather data to be able to answer RQ4 and RQ5. For the questionnaire design, inspiration is also drawn from other large-scale surveys and questionnaire design sources in the software traceability and marketing field.

In addition to the literature review and questionnaire, semi-structured interviews are conducted with several participants of the survey to supplement the results from the literature study and questionnaire to specifically gain a better and more detailed understanding on the perceptions and rationale of practitioners. The reason to opt for a semi-structured interview is so that it leaves room for interviewees to add new insight while it also ensures that the interview stays on course. Depending on the number of interviews and amount of interview data, content analysis will be used to analyze the qualitative data in a more systematic way.

2.1.3 Validation

The goal of this phase is to validate the designed research setup and the designed treatment/measurement specifications and to analyze if these achieve their intended goal or not, and how they could be improved.

For the validation of the questionnaire and its design, there are several phases. First phase consists of several validation sessions that are held with the supervisors in order to refine the initial draft questionnaire (e.g. design and questions) until it is deemed satisfactory for the next step of validation phase. The second phase of the questionnaire validation is performed by gathering feedback from experts, either researchers or practitioners in the field of software traceability to improve the questionnaire. In the third phase, the questionnaire is validated with several native English speakers to ensure the quality and comprehensiveness of the questionnaire before the questionnaire is send out.

For the validation of the questionnaire results and to gain a more detailed understanding from the view of practitioners, a sample of the respondents that are available for an interview will be contacted for semi-structed interviews so that more detailed information can be derived regarding their answers and the context they are in.

Table 1 shows an overview of the relations between the research questions and research methods used to answer the questions.

	RQ 1	RQ 2	RQ 3	RQ 4	RQ 5
Literature review	x	x	x		
Questionnaire				x	x
Data analysis				x	x
Expert interviews				x	x
Content analysis				x	x

Table 2: mapping between research questions and research methods.

The literature review serves as theoretical foundation to conduct the study and to be able to design the questionnaire (RQ1 & RQ2 &RQ3). The questionnaire and its data analysis answer RQ4 and RQ5 from a more quantitative and empirical perspective. The expert interviews and analysis of this data serves to give an additional and more detailed qualitative perspective on the results of the data analysis.

2.2 Research Methods and Techniques

To answer the research questions a variety of research methods will be applied as described in 2.1. This section explains these methods and the rationale behind the decision to employ them in this study.

2.2.1 Literature research

The objective of this literature research is to gain a better understanding and overview of the current body of literature regarding software traceability, its benefits, costs, and challenges, and the role of software traceability in the context of SDLC. In addition, it serves to scope out, identify, and categorize relevant literature for content analysis.

The initial idea for the literature research was to perform a systematized mapping review, which is a more structured literature review in the sense that it partially incorporates elements of a full-fledged systematic mapping review and also elements of a systematic literature review based on the guidelines of Kitchenham, (2004) and K. Petersen, R. Feldt, (2008). The objective of this review was to scope out, identify, and categorize the current existing literature in an exhaustive way based on specified key themes to answer parts of the research questions and to disseminate requirements for the design of the value framework.

However, it quickly became apparent during the pilot of the systemized mapping review that the exhaustive database search approach produced too many search results and possible relevant records and that reviewing all of them would not be feasible considering the time and resource constraints.

Instead, considering that several prominent overview studies such as systematic maps and systematic reviews and roadmaps were found during the previous pilot, these will now be used as the main starting point of the literature review. For this literature review, a mixed search approach of database search and snowballing is used to search and gather relevant literature (Wohlin, 2014). The basic procedure of a snowballing search approach is to find a suitable start set of relevant papers, from which other relevant papers can be derived by analyzing the cited papers in the start set. To complement this approach, a database search is used to gather a start set of relevant overview papers in the area of Software traceability and to find relevant literature that would otherwise be missed by only performing a snowballing approach such as papers on perceived value.

The identified relevant literature is gathered and mapped to an open-ended theme/concept matrix in Excel as shown in table 2, so that the literature can be categorized, sorted, and more easily used to distill a general overview of the current practice. By keeping the matrix open ended, new themes based on new gathered knowledge can be added which will help in iteratively scoping, developing and refining our initial research questions.

Ref#\Theme#	Theme 1	Theme 2	Theme 3	Theme 4	Theme 5
Ref 1	x	x	…		
Ref 2	x		…		x

Table 3: Example theme/concept matrix

For the literature review the following criteria were used:

- Relevant overview or roadmap studies that focus on one of the key themes:
  - Software traceability: studies that give an overview of the fundamentals of software traceability and representative overview studies of the main research fields of software traceability.
  - Empirical studies that study software traceability with a focus on the perspective of practitioners regarding the value, benefits, costs, challenges and needs of Software traceability.

And the following exclusion criteria:

- If it is not in English
- If it is not peer reviewed
- If it is grey literature

Relevant papers are searched by consulting the following databases:

- Google Scholar [GS]
- ACM Digital Library [ACM]
- SpringerLink [SL]
- IEEE [IEEE]

The search terms used to search for relevant literature are shown in table 4:

Search terms
Software traceability AND requirements traceability
Systematic review AND software traceability AND requirements traceability
Perceived value AND measuring AND definition
Software traceability information retrieval
Software traceability approaches
Software traceability AND Agile
Software traceability AND safety critical
Software traceability AND requirements traceability AND survey OR questionnaire
Software traceability AND requirements traceability AND practice OR empirical
Software traceability AND value
Software traceability AND benefits
Software traceability AND costs
Software traceability AND needs
Barriers AND software traceability AND practice
Factors AND software traceability AND practice
Value based software traceability
Feature based software traceability

Table 4: Used search terms

When relevant papers are found, these are snowballed by means of backwards snowballing by scanning the citation list of the papers itself and forwards snowballing by checking the cited by papers list in google scholar.

2.2.2 Questionnaire

The main approach of answering research questions 4 and 5 is with the use of an online questionnaire to collect the necessary data and then to analyze it with the use of statistical methods.

As described in previous sections, there is a lack of empirical data regarding the views of practitioners. With a questionnaire, it is possible to collect large amounts and wider variety of empirical data compared to case studies or interviews. In addition, considering that the research questions are investigating a subjective view of practitioners, questionnaires are a good tool to gather subjective empirical data. Furthermore, the use of a questionnaire minimizes the influence of the researcher on the gathered data since the researcher and respondent will have limited contact.

Using a questionnaire however also has its downsides. One of them is that it will not be possible to go in detail of the decisions made by the practitioners themselves when answering the questionnaire. In addition, questionnaires cannot be too long since a longer questionnaire can negatively impact the response rates. As such, trade-offs must be made between the number of questions and how specific these questions will be.

Therefore, to complement the questionnaire and some of its disadvantageous, semi-structured expert interviews are conducted as well to supplement and enrichen the results.

The design of the questionnaire and the formulated statements are described in further detail in section 4.

As for the target demographic, we look for a broad range of practitioners that have experience and or currently work in the software development, be it as product owner, requirements analyst, developer, tester, and other related roles. Considering it is mainly and opinion-based study, we did not exclude participants that do not perform traceability in their projects. The participants were mainly found via LinkedIn, Reddit, and direct personal contacts.

As for the protocol that the participants follow when filling in the questionnaire, the participants are first presented with the front page which explains the goal of the questionnaire. After that, to make a common ground and understanding regarding the meaning of software traceability, we explain in page 2 what is meant with software traceability to give participants an idea of what the subject is about or to give them a refresher. After that they fill in demographic data of themselves, their company, and their project characteristics. After filling in the demographics, they are presented with statements that are about their current or most recent situation in which they can disagree or agree, which shows us the current state of traceability use, perceived value, and issues in their projects. After that, they are presented statements about possible needs and can indicate the degree of need they have for a specific topic/statement. Depending if contradictory answers are found between current situation and needs, they are presented with an optional page to explain these contradictions. After that they are presented with the last page in which they can fill in extra feedback about the questionnaire or the topic and describe more of their rationale where needed. In addition, they can indicate if they want to do a follow up interview to explain more of their rationale and to learn more

about their project and company environment and the rationale behind their answers in which the interview supplements the questionnaire.

2.2.3 Expert interviews

To supplement the results from the literature research and questionnaire, semi-structured expert interviews are conducted with several respondents of the survey to gain a more detailed understanding in the thought process of the respondents. A semi structured approach is chosen because it lets room for the responder to add extra knowledge that would otherwise be left out compared to a fully structured interview. In addition, it lets the interviewer and interviewee still stay on track by having structure in place and knowing which topics to discuss during the meeting.

As for the target demographic for the interviews, it mainly comprises of people that have filled in the questionnaire so that follow up questions can be asked based on their questionnaire answers. In case there are many participants that are willing to partake in the interviews, an equal number of participants will be chosen between the main categories that are investigated in this study.

Regarding the protocol of the interviews, the interviewer first introduces the goal of the interview. After that, questions are asked to the participants to gain more insight into their personal background and their company background. After that, the interview follows the same structure of the questionnaire, in which the interviewer asks about the rationale and additional information regarding the answers of the interviewees on the questionnaire. While doing so, the answers of the interviewee on the online questionnaire are presented to them as to refresh their memory and help them remember their answers and the rationale they used when answering the questionnaire.

2.3 Research execution plan

A general overview of the research execution is shown in Figure 4, which is an adapted version of Wieringa's empirical cycle. The most important difference is that there will be multiple iterations between the research problem analysis, research and inference design, and validation, before moving on to the research execution. Figure 5 shows a more detailed overview of the steps taken during each phase as described in this chapter.

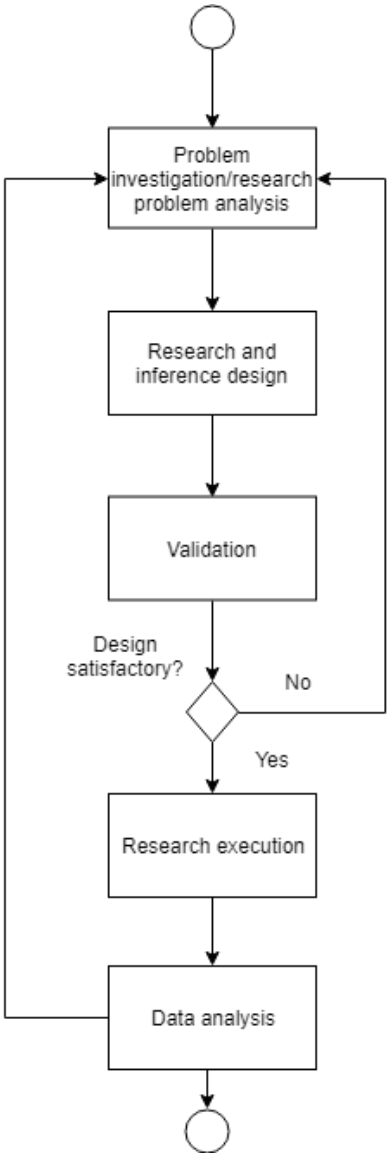


Figure 4: Overview research phases

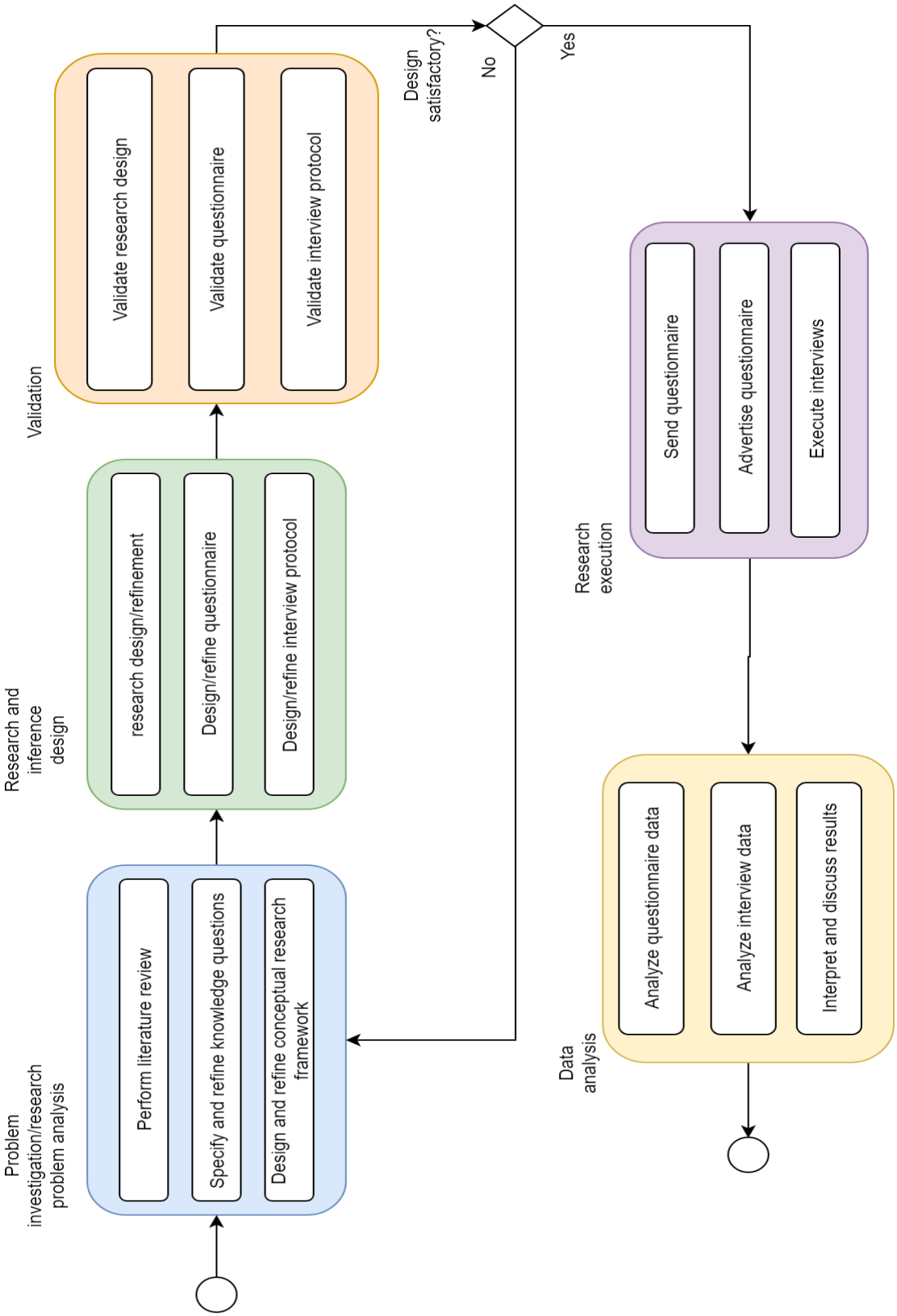


Figure 5: Detailed overview of research execution

3. Literature research

The following section gives an overview of the theoretical background of software traceability. The first subsection touches upon the key terminology used by the software traceability community and this study. The second subsection describes the basic traceability process. The third section gives an overview of the history and current state of software traceability research. The fourth section gives an overview of some of the more notable and influential empirical studies conducted. The fifth section touches upon the differences between software traceability in Agile and Traditional environments. The sixth section gives a summary of the benefits and costs of software traceability in practice and the current state of research surrounding it. Finally, the last section summarizes and discusses some of the main findings which serves to design and scope the more empirical part of this study.

3.1 Key terminology of Software Traceability

The following section touches upon the important basic, fundamental, and key terminology of software traceability and the associated tasks to provide a common ground to build upon and to be able to understand software traceability. Most of the following information about the basics of software traceability is a summarization of the necessary information which is derived from the book Software and Systems traceability by Cleland-Huang et al., (2012). In this book they provide a mostly complete and comprehensive overview of the essential terminology regarding software traceability that has been developed and endorsed by the members of the traceability community, which is used throughout this study. For more detailed information, clarifications or examples it is strongly suggested to read the book.

According to Cleland-Huang et al., (2012) and as described in section 1, in its elementary form, software traceability is the potential to relate different data stored in a variety of artifacts of some kind, and the ability to examine these relations. The value of traceability is that the use of these relations can enable many software and systems engineering activities and tasks, such as change impact analysis, coverage analysis, dependency analysis, and many more. As such, traceability can provide visibility and a better understanding of the software system under development.

The two most fundamental building blocks for software traceability are the trace artifacts and trace links. Trace artifacts are essentially traceable units of data that are present in a development project. This can refer to single requirements, to clusters of requirements, the whole document that contains all the requirements, to classes, methods, lines of code, and all other possibly produced artifacts in a software development project that contains data of any sort. To keep it simple on a conceptual level, this study follows the definition of trace artifacts coined by Cleland-Huang et al., (2012), “in which artifact refers to both the objects, documents as a whole and to any internal delineation therein”.

The second fundamental building block are the *trace links*. According to Cleland-Huang et al., (2012) a Trace link is:

> *Trace link – a single specified association/relation between a pair of artifacts, one compromising the source artifact and one compromising the target artifact. The trace link is one of the trace elements. It may or may not be annotated to include information such as the link type and other semantic attributes. This definition of trace link implies that the link has a primary trace link direction for tracing. In practice, every trace link can be traversed in two directions, A to B and B to A, so the link also has a reverse trace link direction for tracing. The trace link is effectively bidirectional. Where no concept of directionality is given or implied, it is referred to solely as an association.*

Based on the definition of trace link according to Cleland-Huang et al., (2012), a *trace link* is in essence an association or relation between a pair of trace artifacts, the source and target artifacts, which for

example could be a requirements document in relation to a code file. It is implied that the link can be traversed in different directions, in a forward fashion, from source to target which is the *primary trace link direction* and from a backwards fashion from target to source which is called a *reverse trace link direction*, and *bi-directional* in which it is possible to traverse a trace link in both a forwards and backwards fashion or in other more correct terms in a primary trace link direction and in a reverse trace link direction.

According to Cleland-Huang et al., (2012), the term *trace relation* is used interchangeably with the term *trace link* in many publications. This is not a surprise considering that terms such as trace link and trace relation could seem to mean the same thing and could be perceived as ambiguous to some people. As such, to encourage the more consensual use of terminology in the traceability community they have made the following distinction between *trace relation* and *trace link*:

> *Trace relation – all the trace links created between two sets of specified trace artifact types. The trace relation is the instantiation of the trace relationship and hence is a collection of traces. For example, the trace relation would be the actual trace links that associate the instances of requirements artifacts with the instances of test case artifacts on a project. The trace relation is commonly recorded within a traceability matrix.*

As Cleland-Huang et al., (2012) have defined the distinction between *trace relation* and *trace link*, a *trace relation* is in essence the collection of trace links between two types of trace artifacts. To visualize, look at figure 6. Usually there are more than one trace link possible and needed between two trace artifacts.

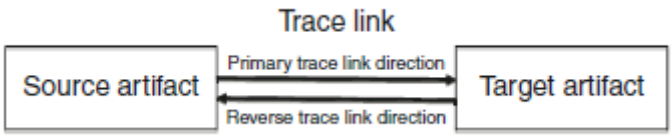


Figure 6: trace relation and trace link (Cleland-Huang et al., 2012).

Touching back on the point of ambiguity of certain terms used throughout the traceability community, the term *trace* itself has led to some misunderstandings since it has two distinct meanings depending if the term is used as a noun or as a verb (Cleland-Huang et al., 2012). In the case of a noun, it denotes e.g. a trace link or basically a mark or relation available between two artifacts while the verb means the actual act of tracing, tracking, or following a e.g. a trace link. Because the noun and verb version of *trace* have been used interchangeably in many publications, even though they essentially have different meanings, they have defined the following two distinct definitions of *trace*:

> *Trace (noun) – A specified triplet of element comprising: a source artifact, a target artifact and a trace link associating the two artifacts. Where more than two artifacts are associated by a trace link, such as the aggregation of two artifacts linked to a third artifact, the aggregated artifacts are treated as a single trace artifact. The term applies, more generally, to both traces that are atomic in nature or chained in some way.*
>
> *Trace (verb) – The act of following a trace link from a source artifact to a target artifact (primary trace link direction) or vice-versa (reverse trace link direction).*

Based on their definition, a trace basically refers to the triplet of basic building blocks of traceability which is comprised of a source artifact, target artifact, and the associated trace link between the two artifacts. In addition, such a trace can be either *atomic* or *chained*, in which an atomic trace refers to

the trace between e.g. artifacts A and B, and a chained trace refers to the trace between artifacts A and B, but also any other subsequent traces with other artifacts e.g. a chained trace would refer to the trace established between A, B, and C. See figure 7 for an exemplary visualization.

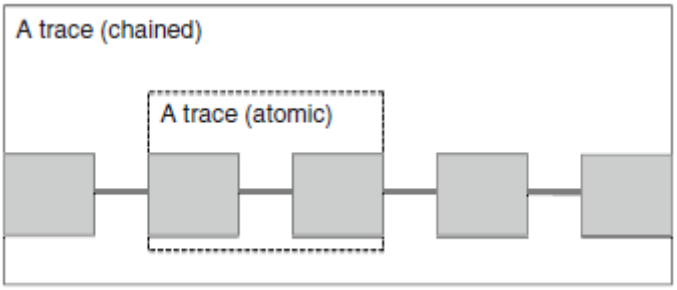


Figure 7: atomic vs chained trace (Cleland-Huang et al., 2012).

The term *traceability* is defined as:

> *Traceability* – the potential for traces to be established and used. Traceability (i.e., trace "ability") is thereby an attribute of an artifact or of a collection of artifacts. Where there is traceability, tracing can be undertaken, and the specified artifacts should be traceable.

The ability to trace, traceability, is thus the ability and possibility to be able to establish a trace between trace artifacts. Considering there are many possible types of trace artifacts that can be present in a project depending on a variety of contextual factors, there are several common reoccurring terms to denote and delineate certain types or forms of traceability based on the artifact types. These terms include Requirements traceability, Software traceability, and systems traceability.

> 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 periods of ongoing refinement and iteration in any of these phases)."
>
> Software traceability – Extending the definition to encompass and interrelate any uniquely identifiable software engineering artifact to any other.
>
> Systems traceability – Extending the definition of requirements traceability to encompass and interrelate any uniquely identifiable systems engineering artifact to a broad range of systems-level components, such as people, processes and hardware models.

Tracing implies undertaking all those activities required to put traceability in place and demands some form of agency. This led to the terms of manual, automated, and semi-automated tracing which is defined by Cleland-Huang et al., (2012) as:

> Manual tracing – When traceability is established by the activities of a human tracer. This includes traceability creation and maintenance using the drag and drop methods that are commonly found in current requirements management tools.
>
> Automated tracing – When traceability is established via automated techniques, methods and tools. Currently it is the decision as to among which artifacts to create and maintain trace links that is automated.

Semi-automated tracing – When traceability is established via a combination of automated techniques, methods, tools and human activities. For example, automated techniques may suggest candidate trace links or suspect trace links and then the human tracer may be prompted to verify them.

3.2 The basic software traceability process

(Cleland-Huang et al., 2012) distinguishes the process of traceability between four fundamental tasks, which are:

- Planning and Managing Traceability Strategy
- Creating traces
- Maintaining traces
- Using traces

Figure 8 displays a generic traceability model. This model serves to give an overview of the basic processes and responsibilities associated with traceability.

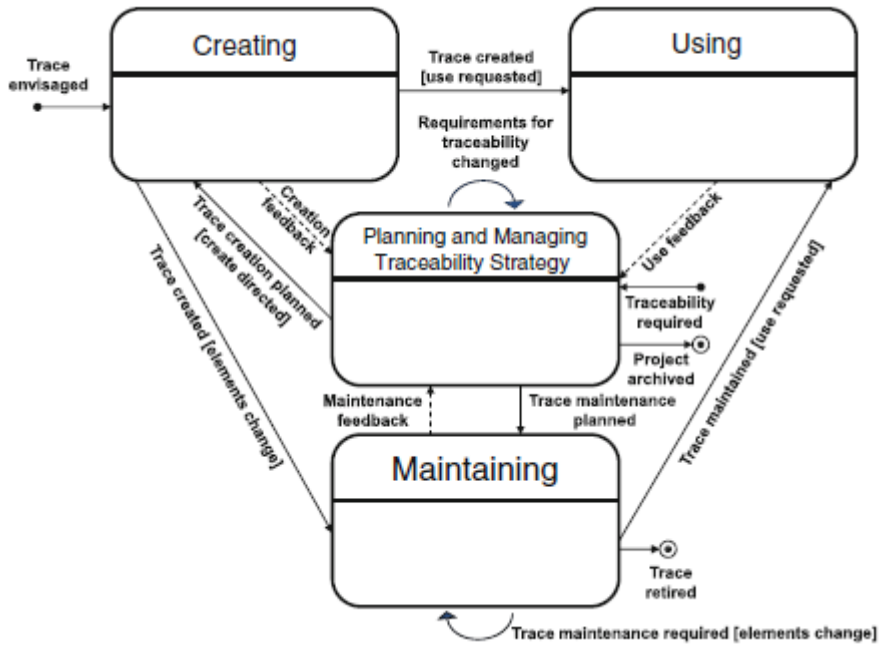


Figure 8: A generic traceability model (Cleland-Huang et al., 2012).

3.2.1 Traceability Strategy

According to (Cleland-Huang et al., 2012), effective traceability rarely happens by chance or through ad hoc efforts. One of the important and initial tasks for a project is to put a planning and managing strategy in place for traceability, which is defined by the authors as:

traceability strategy – Those decisions made in order to determine the stakeholder and systems requirements for traceability and to design a suitable traceability solution, and for providing the control necessary to keep these requirements and solutions relevant and effective during the life of a project. Traceability strategy comprises traceability planning and traceability management activities.

A generic traceability strategy model is shown in figure 9.

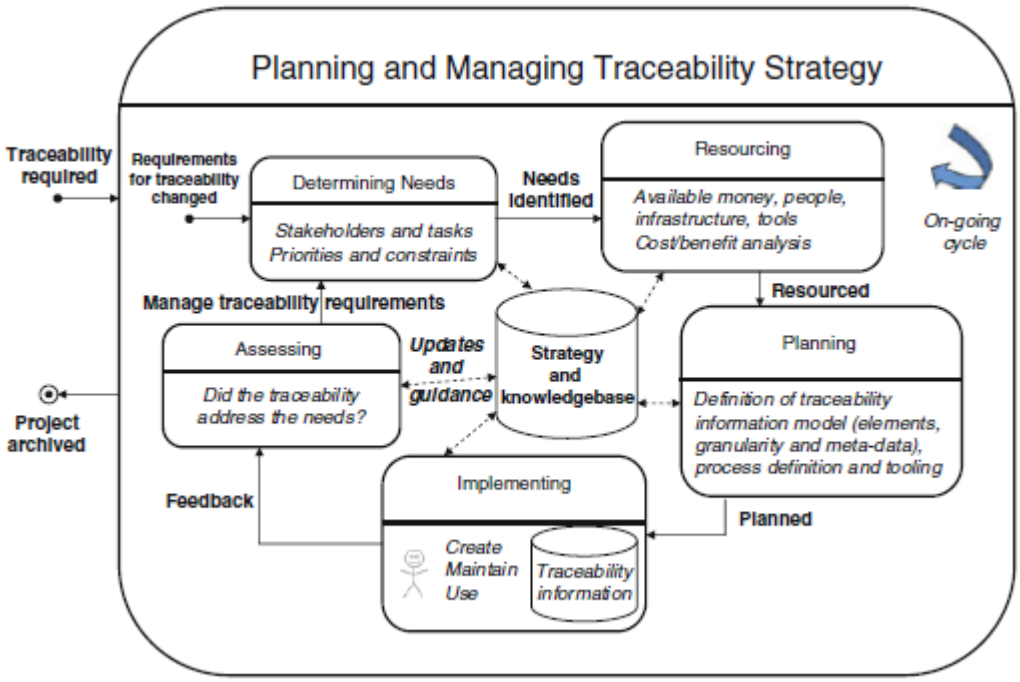


Figure 9: A generic traceability strategy model (Cleland-Huang et al., 2012).

The traceability strategy includes:

- Determining the needs of the project depending on stakeholders, tasks, priorities and constraints.
- Managing the resources available, such as available money, people, infrastructure, tools, and performing cost/benefit analyses.
- Planning, as in which artifacts to trace and in what granularity.
- Implementing, as in executing the plan and creating, maintaining, and using the traceability information during the project.
- Assessing, as in checking if the traceability information and associated tasks addressed the needs of the project and stakeholders.

As such, traceability is mostly a supporting system that is concerned with answering specific project specific questions of the associated stakeholders (Cleland-Huang et al., 2012). This means that it is rather unfeasible to have a general traceability strategy in place that would work for every project considering that the needs, resources, and available data varies on a project by project basis. Not only is this the case, it is also cost-ineffective to have a traceability solution in place for a specific project that answers to all the needs of that project because they usually have limited resources available. Therefore, determining whose needs to satisfy, which traceability tasks and activities to enable, and what the constraints are, is a value decision and trade-off that lies within each traceability strategy (Cleland-Huang et al., 2012).

Not only is it important to initiate a traceability strategy, it is also important to manage it and adapt it over time. The needs, resources, constraints, and available data are not set in stone from the beginning of the project and evolve over time. Therefore, it is also important to manage the traceability strategy over time and to assess the quality and execution of the solution by having a feedback loop in place that makes use of historical data (Cleland-Huang et al., 2012).

To have an overview of all the possible traces, these are usually provided in an overview or traceability information model, also referred to as TIM's. TIMs are graphs that:

> *Traceability Information Model* -- defines the permissible trace artifact types, the permissible trace link types and the permissible trace relationships on a project, in order to address the anticipated traceability related queries and traceability-enabled activities and tasks.

Figures 8 and 9 are examples of TIM models.

3.2.2 Creation

The second fundamental task is the creation of traceability or in other words according to (Cleland-Huang et al., 2012):

> Traceability creation -- the general activity of associating two (or more) artifacts, by providing trace links between them, for tracing purposes. Note that this could be done manually, automatically or semi-automatically, and additional annotations can be provided as desired to characterize attributes of the traces.

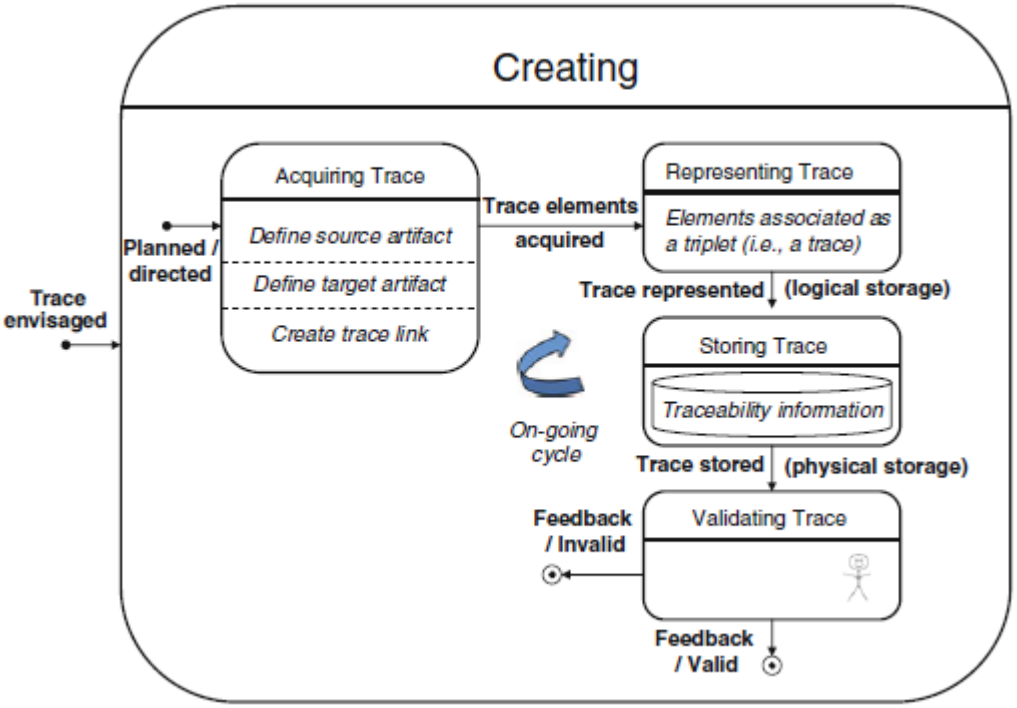


Figure 10: A generic trace creation process (Cleland-Huang et al., 2012).

Figure 10 shows the generic process model for creating traceability which constitutes of

- Acquiring trace, which includes defining source artifact, target artifact, and the trace link.
- Representing trace, how the trace is represented and stored logically.
- Storing trace, how the trace data is stored physically.
- Validating trace, validating if the trace is valid or invalid.

Software artifacts typically already exist in a project, but trace links may not yet be defined (Cleland-Huang et al., 2012). The creation of traces is typically categorized in two categories, which are named

trace capture and trace recovery. With trace capture, the traces are created in a forward fashion, which means that they are created and evolved concurrently with the artifacts and the forward engineering process. With trace capture, the traces are created in a later time where for example certain engineering processes are already completed but where they still need traceability information from.

In addition, traces can be created manually, semi-automatic, or automatic. No matter which of the approaches is used. The validation is critical for determining and assuring the credibility of a trace as a whole (Cleland-Huang et al., 2012). As such, not only much of the current traceability research focuses on the creation of traces, it also focuses on ways to validate the traces, which could take forms of improving the accuracy of automated approaches to tasking people to validate the created traces.

3.2.3 Maintenance

The third fundamental task is the maintenance of traceability, which is described as:

> Traceability maintenance -- associated with activities that update preexisting traces as changes are made to the traced artifacts and the traceability evolves, creating new traces where needed to keep the traceability relevant and up to date.

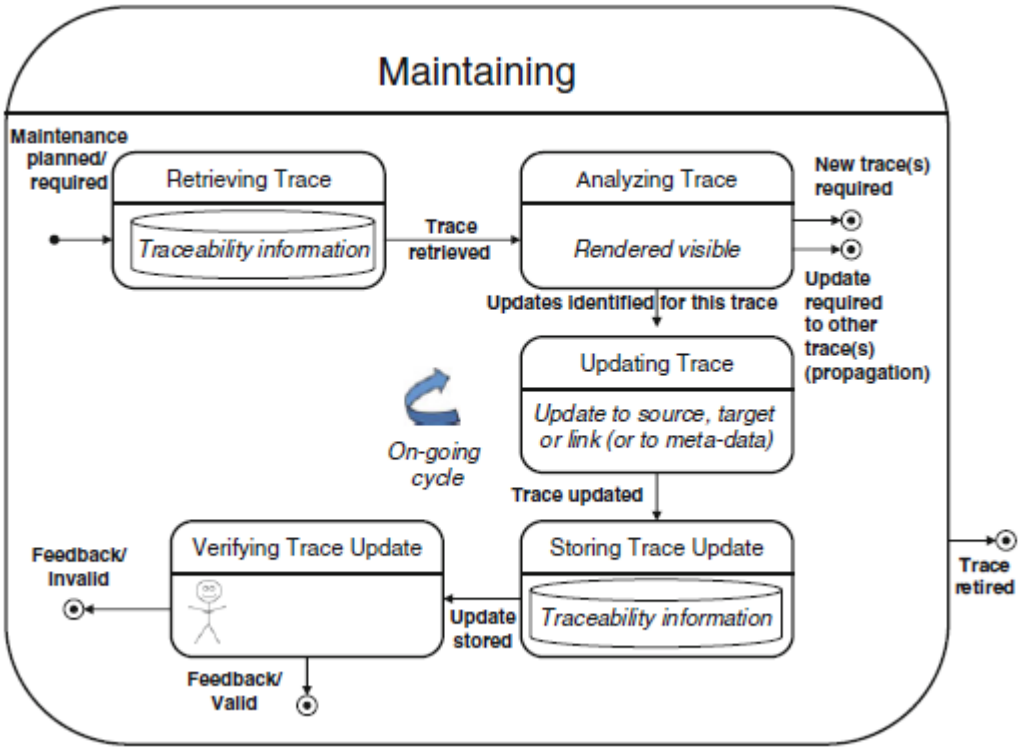


Figure 11: A generic trace maintenance process (Cleland-Huang et al., 2012).

Figure 11, shows the generic process model for traceability maintenance which constitutes of:

- Retrieving trace
- Analyzing trace
- Updating Trace
- Storing Trace update
- Verifying Trace Update

The traces that are made in a certain point of time have to be maintained and updated over the life span of a project. Changes that can trigger the maintenance can come from a variety of sources such as changes in the underlying artifacts, changes in the needs of the stakeholders, and changes to the overall traceability strategy (Cleland-Huang et al., 2012). The maintenance of traces depends on what update is necessary, but it could mean the creation of entirely new traces or updates to parts of the current traces. The maintenance can be done in a continuous or on-demand fashion just like the creation of traces.

3.2.4 Use

The fourth fundamental task is the use of traceability:

> Traceability use -- those activities associated with putting traces to use to support various software and systems engineering activities and tasks, such as verification and validation, impact analysis and change management.

Figure 12 show the generic process model for using traces which constitutes of:

- Retrieving trace
- Rendering Trace
- Assessing Trace
- Recording trace use

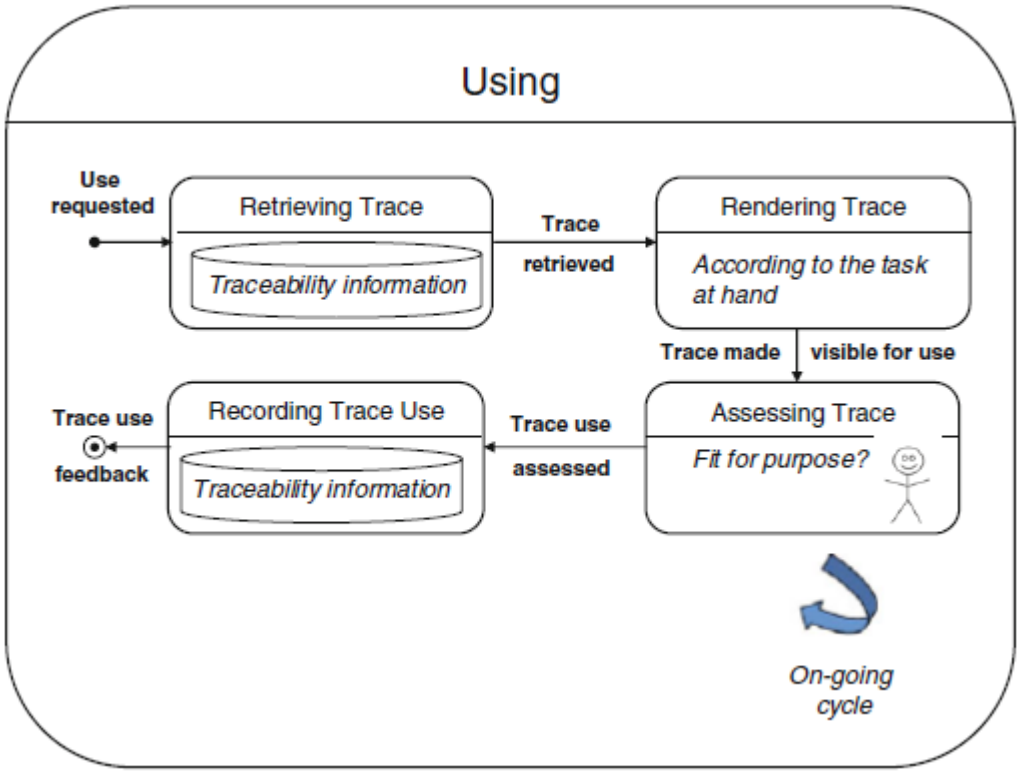


Figure 12: A generic trace use process (Cleland-Huang et al., 2012).

Every atomic trace could play a role in many different use contexts. It could be used in isolation or together with other traces in a chain. To be able to use these traces, it is important to be able to retrieve, rendered, assess, and record the traces and their uses. The use of traces can be categorized as short-term usage and long-term usage. Typical examples of short-term uses of traceability include requirements completeness analysis, requirements trade-off analysis or requirements-to-acceptance-

test mapping for final acceptance testing. Typical long-term uses include determination of effects of changes to a software system or the propagation of changes during its evolution.

3.3 History and current state of software traceability research from a bird's eye view

The following sections gives a brief overview of the history and current state of research of software traceability.

Software traceability is a large research field that has been researched for over several decades. However, despite the significant advances since one of the earliest tool and process support for traceability in practice in the 1970s (Pierce, 1978), there were still many challenges surrounding software traceability in the 2000s in regards to implementing successful and cost-effective traceability.

To combat these challenges, software traceability researchers have organized workshops with academic, government, and industrial researchers and practitioners over the past two decades to articulate the challenges and to established a set of challenges to systematically solve (Orlena Gotel et al., 2012). These challenges are described as the '*grand challenges of traceability'*, and are periodically assessed by the software traceability researchers to gain an overview of the progress on these challenges (Orlena Gotel et al., 2012; "Grand Challenges of Traceability 2017," n.d.).

By solving these challenges, the traceability community hopes to achieve their vision of how traceability should support the development practices in 2035. The vision for traceability in 2035 is that traceability is ubiquitous in the development, meaning that it is in the background, trusted by all stakeholders, and completely requirements driven (Orlena Gotel et al., 2012). As how Orlena Gotel et al., (2012) have summarized this vision:

> " *Traceability will be the thread that weaves data together on a project to tell a myriad of stories, from the rationale underlying decisions through to the underlying social network that came together to make these decisions and is, therefore, best able to change the, Traceability will be completely requirements-driven in 2035*".

To achieve this vision of traceability, they have based and categorized the challenges of traceability around several assumptions that must be demanded of traceability in practice in 2035. These are that in 2035, traceability is assumed to be (Orlena Gotel et al., 2012):

1. Purposed. Traceability is fit-for-purpose and supports stakeholder needs (i.e., traceability is requirements-driven).
2. Cost-effective. The return from using traceability is adequate in relation to the outlay of establishing it.
3. Configurable. Traceability is established as specified, moment-to-moment, and accommodates changing stakeholder needs.
4. Trusted. All stakeholders have full confidence in the traceability, as it is created and maintained in the face of inconsistency, omissions and change; all stakeholders can and do depend upon the traceability provided.
5. Scalable. Varying types of artifact can be traced, at variable levels of granularity and in quantity, as the traceability extends through-life and across organizational and business boundaries.
6. Portable. Traceability is exchanged, merged and reused across projects, organizations, domains, product lines and supporting tools.

7. Valued. Traceability is a strategic priority and valued by all; every stakeholder has a role to play and actively discharges his or her responsibilities.
8. Ubiquitous. Traceability is always there, without ever having to think about getting it there, as it is built into the engineering process traceability has effectively: “disappeared without a trace”.

Even though all of these are significant challenges, the 8th challenge is commonly referred to as the *Grand Challenge of Traceability*. The reasoning behind it is, to achieve ubiquitous traceability, all the before mentioned challenges must be addressed. Associated with the grand challenge, Orlena Gotel et al., (2012) have stated the following long research theme: “To provide automation such that traceability is encompassed within the broader software and systems engineering processes, and is integral to all tool support”.

Since the establishment of these challenges, much of the software traceability research has been able to focus or can at least identify itself with one or several of these challenges. It is important to note, that all these challenges have cross cutting concerns and thus, by tackling a specific challenge it is most likely that parts of other challenges are addressed as well.

Much progress has been made in many research areas denoted by the first Grand Challenges document as described before (Antoniol et al., 2017). In 2017, several of the researchers that organized the first grand challenges workshops in 2006 came together again and organized another session with researchers and practitioners to understand the more recent progress and challenges in the research surrounding software traceability. This session resulted the second grand challenges document:” Grand Challenges of Traceability: The Next Ten Years”.

According to the authors, which is also noticeable when navigating the software traceability literature, a big focus of the more recent research is focused around the automation of the traceability process, which also relates to the long research theme as stated before (Antoniol et al., 2017; O. Gotel et al., 2012; Orlena Gotel et al., 2012). One of the key drivers for this focus is that, in a traditional sense, the effort for establishing traceability is greater than the benefits in practice (Arkley & Riddle, 2005; Ingram & Riddle, 2013). As such, many researchers try to come up with methods to automate parts of the traceability process or try to improve these methods to reduce the effort needed to create and maintain trace links.

Much of the automation research regarding trace creation and maintenance focuses on the more textual ontology and information based retrieval approaches (Antoniol et al., 2017). These use techniques and methods such as Natural Language Processing (NLP) and deep learning to automatically create and maintain traces based on textual relations between the artefacts and to improve the efficiency and accuracy of these techniques (Guo, Cheng, & Cleland-Huang, 2017; Lam, Nguyen, Nguyen, & Nguyen, 2016; Zhang, Witte, Rilling, & Haarslev, 2008; Borg, Runeson, & Ardö, 2014; Dekhtyar, Poly, Obispo, & Hayes, 2018). Besides this more mainstream approach, other information sources of systems are also used to create and maintain trace links, such as utilizing source code change patterns to evolve trace links across different software versions and by analyzing and using eye tracking data of practitioners (Antoniol et al., 2017).

Other areas discussed in the second Grand Challenges document are trace strategizing, trace link usage, real-world applications of traceability, and traceability datasets and benchmarks. (Antoniol et al., 2017). Several challenges in these areas that they discussed and need more research on are the challenges surrounding publicly available datasets, challenge of benchmarking in traceability, how to create a strategy and the tradeoff between generalizability and customizability, formal and less formal traceability information models depending on the application domain and development

processes for example between safety critical projects and agile projects, what the cost-benefits considerations should be of these strategies, the adoption of theoretical approaches in practice and also that even though there has been progress in studies on the usage of traceability and traceability links, more research is still needed in the area of traceability usage and the other areas surrounding software traceability.

Now that we have gone through some of the theoretical and research background of software traceability from a bird's eye view, the next sections details some of the more influential empirical based studies that gives us more information about the history and current state of software traceability from the practitioner's perspectives.

3.4 Software traceability in practice

The following section gives an overview of the related work regarding some of the notable and influential empirical studies studying the stakeholder perceived value perceptions of software traceability and some of the major factors and problems found by these studies that influence the use and adoption of software traceability in practice.

Compared to the numerous amounts of research papers that are available that focus on tools, methods, and techniques found during the literature search, there are fewer papers that address traceability from the view of the users and stakeholders in practice. Even though they are in fewer numbers, the results of these studies are of paramount importance considering much of the traceability research is driven by the needs of practitioners (Orlena Gotel et al., 2012).

One of the first major studies related to the view of users and stakeholders in practice was conducted around two decades ago by Gotel and Finkelstein in the mid-1990s (O. C. Z. Gotel & Finkelstein, 1994). Their focus was to investigate and discuss the underlying nature of the requirements traceability problem, in which their work is based on empirical studies with over 100 practitioners. In this study they suggest the distinction of two types of requirement traceability types to understand the problem, *pre-requirements specification (pre-RS)* and *post-requirements specification (post-RS)* with the following definition:

- "Pre-RS traceability, which is concerned with those aspects of a requirement's life prior to its inclusion in the RS (requirement production)."
- "Post-RS traceability, which is concerned with those aspects of a requirement's life that results from its inclusion in the RS (requirement deployment)."

The main distinction between these types is the information that they deal with. Pre-RS is concerned with information and the traceability of requirements regarding the sources and origin (e.g. stakeholders) of a specified requirement while post-RS is involved with the traceability of a specified requirements through the chain of artifacts in which the requirements are distributed.

One of the factors that the authors identified for the poor and varying requirements traceability in practice is due to lack of or inadequate pre-RS traceability support and the so called *establish and end-use conflict.* With this they mean that the two parties involved, those in a position to make it possible and those who require it to assist their work, have conflicting problems and needs, making it difficult to satisfy both parties. They describe that technology alone will not provide a solution for traceability and that it is also important to look at the social aspects of the activities.

In addition, the authors reported that the inability to locate and access the sources of requirements and pre-RS work was the most commonly cited problem across all the practitioners in their investigation, which is seen as a major contributor to other traceability problems. The reasons for this

problem were twofold. The first reason was due to politics, which prohibited any knowledge of, or access to, the original sources or requirements engineers. The second reason behind this problem was reported to be the difficulty in keeping track of the original sources and subsequent traces of participation.

Furthermore, certain project characteristics were found that increase or decrease the occurrence of the localization and access problem. One of the characteristics that increased the occurrence was that the individuals of a project were split across several teams. This made the location and access of sources difficult because of a lack of shared or project-wide commitment; information loss; inability to assess the overall state of work or knowledge; little cross-involvement; poor communication; minimal distribution of information; and changing notions of ownership, accountability, responsibility, and working structure. Characteristics that reduced its occurrence were found in projects consisting of few individuals, due to a clear visibility of responsibilities and knowledge areas; clarity of working structures; team commitment and ownership; and individuals who acted as common threads of involvement.

The second large survey investigating traceability in practice was performed by Ramesh in the late 1990s (Ramesh, 1998). In this study the author focused on factors that influence the requirement traceability in practice. The author investigated the role of institutional context and the strategic conduct in explaining differences in traceability practice across system and development efforts. The author studied how environmental, organizational, and system development context factors influence the adoption and use of requirements traceability by contrasting two extremes of traceability users referred to as *low-end and high-end* users. The views and practices regarding traceability differed significantly, where on the low-end users see traceability as a mandate and implement simple traceability tactics to address their needs. On the high end, the users see traceability as an important component of their development and implement and use more advanced and richer traceability tactics.

Furthermore, the author discovered that the managers of low-end users claim that they do not derive much benefit from their simple traceability practices and that it is viewed as a 'necessary evil'. According to Ramesh, this severely limits the usefulness of their traceability practices. In contrast, the author discovered that managers of high-end users are committed to traceability and see strategic benefits of incorporating traceability practices, even when it is not required.

Additionally, Ramesh reported that there is a lack of interest in traceability among low-end developers. This was explained by the lack of organizational commitment, management support, adequate tools, methodologies, and training. In contrast, the authors described that high-end developers have a variety of motivating factors to view traceability in a positive light, such as adequate incentives, atmosphere of knowledge sharing and growth and organizational support.

In 2005, Arkley and Riddle performed a survey by means of questionnaires and interviews of nine software projects ranging in complexity (Arkley & Riddle, 2005). Their observations were in line with the observations from similar studies in the 1990s. Furthermore, the authors argue that the poor traceability practices are due to the traceability benefit problem:" the lack of direct, tangible benefits to the main development process from traceability". This problem is in a certain sense similar to the conflicts in need between the provider and user of traceability as identified by Ramesh, (1998). However, the traceability benefit problem focuses on that the one creating and maintaining the trace links does not see direct benefits of tracing to the goals of their own tasks and therefore can treat traceability as an extra overhead considering the amount of effort is required to establish and maintain traceability.

In 2007, Blaauboer, Sikkel, and Aydin conducted a case study in a large IT company to identify factors that are relevant for the decision to adopt traceability, or not, in an information systems development project (Blaauboer et al., 2007). From this study, five dominant factors emerged that influenced the adoption of traceability: development organization awareness, customer awareness, return on investment, stakeholder preferences, and process flow. One of the significant findings was that it turned out that most of the software development project leaders that they interviewed were not aware of the concept of traceability. As a result, traceability in some of the software projects were not even considered, which serves to intensify the big differences in traceability adoption and use in practice.

In 2009, Mäder, Gotel, and Philippow conducted a practitioner survey to get a high-level update on the traceability practices and problems (Mäder et al., 2009). As result of this study, the authors identified the importance of prevailing motivation and underlying traceability in an organization. They characterized these motivations based on four environments:

1. Regulated – to satisfy industry-imposed or company-imposed regulations
2. Sub-contractor – to align with the processes and demands of customers
3. Consultant – to support internal or external company-specific efforts to improve processes
4. Enthusiast – due to an energized individual with knowledge and a passion to learn.

In addition, the problems that they observed were like the previous described studies. Similar to the findings of Ramesh, (1998), they identified two distinct groups of users. The distinction, however, was more due to motivation and organizational setting compared to the intended use and practice of traceability. Nonetheless, the multiple expectations of what traceability should do at a project level was still present and the actual direct benefits seemed to be still a problem.

Moreover, the authors described that there was a need, but almost no guidance available, for practitioners to help them establish traceability in projects but also across project, organizational, and regional boundaries. Also, the authors noted that the role of tools became more important in practice, in which the increasing complexity of systems played a role.

In 2013, Bouillon, Mäder, and Philippow, conducted a survey that helped the traceability community understand which traceability usage scenarios were most relevant for practitioners (Bouillon et al., 2013). For this study, the authors reviewed the existing literature to draw up an initial catalog of typical usage scenarios of software traceability in practice which is meant to enable their study but also as a possible reference source for future studies. Resulting from this study, it was identified that most of the practitioners used one or more of the commonly reported usage scenarios of software traceability. The most frequently used usage scenarios were finding origin and rationale of requirements, documenting a requirement's history, and tracking requirement or task implementation state.

Nonetheless, the authors still emphasized that very little is known about practitioners use and need regarding software traceability and the importance of performing more empirical studies. The authors also observed the traceability benefit problem, in which the practitioners commented that they struggle with the bad cost-benefit ratio.

In 2018, Wohlrab et al., conducted a multiple case study with 24 individuals from 15 industrial projects (Wohlrab et al., 2018). The authors investigated the collaborative aspects of traceability from the perspectives of organization, process, and culture. The key challenges they explored were how traceability management can support collaboration, how collaboration relates to traceability management approaches, and what characteristics of the development effort influence traceability management and collaboration. As result the authors found that the practitioners struggled with (1)

collaboration across team and tool boundaries, (2) conveying the benefits of traceability, (3) traceability maintenance, and that by addressing these challenges it could facilitate collaboration in distributed contexts in which collaboration and traceability management could have the potential to be mutually beneficial. As can be seen from the challenges, these were similar to the problems observed in previous studies, in which practitioners struggled with boundaries, the benefits, and the efforts to maintain traceability.

According to all these studies, there are a myriad of factors that influence the adoption and use of software traceability in practice. Not only are the needs of traceability different per project, it is also influenced by other factors such as the view and awareness of stakeholders, organizational factors, tool support, the cost-benefit problems and lack of guidance. Because of all the varying factors, Regan, McCaffery, McDaid, and Flood, (2012), performed a systematic literature review on the barriers of traceability in practice and their potential solutions based on the empirical studies performed before 2012. Even though, the evidence used in this study dates back to over a decade ago, some of the barriers indicate to be still relevant in the current day and age, as can be seen in the study of Wohlrab et al., (2018). Regan et al., (2012) categorized the barriers as shown in table 5.

Category	Barriers
Management issues	Cost
	Lack of guidance
	Return on investment
	Traceability decay
	Data collection
Social issues	Different stakeholder viewpoints
	Internal politics
	Lack of communication/understanding
Technical issues	Issues with tools
	Storage and versioning
	Complexity

Table 5: Barriers of traceability by Regan, McCaffery, McDaid, and Flood, (2012)

The categorization made by Regan, McCaffery, McDaid, and Flood, (2012), which is based on some of the previous described empirical survey studies, gives us a clear overview of some of significant but also general factors that are the cause for the varying degree of adoption, use, and implementation of software traceability in practice. The authors note that these are relevant in both safety- and non-safety critical projects, but that there are additional issues with tool support regarding automation in the safety-critical projects. The reason for this, is that in such safety-critical projects and consequently heavy regulated areas, accountability is important. A popular example to exemplify this, is the ethical and social discussion around the self-driving cars, where questions are raised around the responsibility and accountability during an accident, which has implications for the parties involved and external parties such as insurance companies. Even though this example is not one-to-one with the complexities perceived in safety-critical projects regarding software traceability, the discussion around automation and accountability can influence the needs of software traceability in heavy regulated areas. Therefore, according to the authors, it might not be practical to automate the whole traceability process in such areas and more semi-automated approaches would possibly be more suitable.

As described in section 3.3**,** all the varying problems identified in empirical studies has driven the software traceability community to tackle these challenges together which resulted in various workshops between researchers and practitioners that led to the grand challenges of software

traceability. Much progress has been made regarding software traceability research, especially in area of automation and tools. In addition, several experimental studies have shown that software traceability can positively impact software engineering tasks directly, such as the experiment of Mäder and Egyed, (2015) in which they conducted a controlled experiment with more than 50 subjects performing maintenance tasks. Their finding was that traceability navigation had a positive effect on the performance, the quality, and workflow of how change tasks were performed, in which their subjects used traceability mainly for navigation through source code and to identify which parts should be changed.

To summarize this section on software traceability in practice, despite the many advances in research regarding software traceability, many influential software traceability researchers still emphasize that there is a long way to go and that there is still a need for more empirical evidence regarding the problems and state of software traceability in practice. In addition, much of the empirical studies in the past indicate to be based on the more traditional software development life cycles. Several researchers have emphasized the need to do more empirical research in the context of different domains outside the usual safety-critical and heavy regulated industries regarding software traceability (O. Gotel et al., 2012). This of course, introduces a myriad of other problems considering that the development context varies even more outside such industries. Nonetheless this has not stopped the software traceability community to research outside the typical domain of safety-critical projects, which is described in the next section.

3.5 Software traceability in the Agile Software Development Life Cycle

As described in the previous section, much of the empirical research on the perspective from practitioners regarding software traceability in the past has been focused around highly regulated and safety-critical industries, where there is a preference for the more traditional software development life cycles. As consequence, much of the traceability research and solutions has focused on the traditional software development where there is a focus on heavy documentation and where phases are planned and conducted sequentially.

Compared to the traditional models such as waterfall, Agile development methods such as scrum, Extreme Programming (XP) and Kanban are driven by the Agile principles as stated in the Agile manifesto where (Agility Alliance, 2011):

- Individuals and interactions are valued over processes and tools
- Working software is valued over comprehensive documentation
- Customer collaboration is valued over contract negotiation
- Responding to change is valued over following a plan

It is stated that, even though there is value in processes and tools, comprehensive documentation, contract negotiation, and following a plan, more value is placed on individuals and interactions, working software, customer collaboration, and responding to change.

Which are based on the following principles in the Agile manifesto:

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of week to a couple of months, with a preference to the shorter timescale.
- Businesspeople and developers must work together daily throughout the project.

- Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Agile processes promote sustainable development.
- The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility
- Simplicity – the art of maximizing the amount of work not done – is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

While there are varying differences between these Agile methods, with the emergence and increasing maturity and adoption of Agile practices there is an increasing imbalance and gap between the research regarding software traceability and the software engineering processes in practice, where Agile developments methods are taking over, even in safety-critical domains (COLLAB.NET & VERSIONONE.COM, 2018; Mc Hugh, Mc Caffery, & Casey, 2012).

The increasing imbalance was not left unnoticed by the software traceability community. Cleland-Huang, (2012) discussed the importance of traceability in agile software development and explored some of the issues, challenges, and goals of traceability in an agile environment. Because Agile projects vary heavily in shapes and sizes, which in turn impacts the needs of traceability, the author suggested the classification of agile projects regarding traceability in:

- small to medium sized agile projects
- large scaled, distributed, or long-lived projects
- and safety-critical and non-safety-critical projects

The author noted that the goals of traceability in some of these environments are like traditional environments, especially in the larger scaled and or safety critical projects, but that the means of achieving these goals in traditional environments must be adapted for agility. This means that agile environments can also benefit and use software traceability for tasks such as change impact analysis, product conformance, and process compliance. However, because agile environments are typically characterized as having short iteration cycles, focus on collaboration, and a focus on working products and consequently on code based on the previously mentioned agile manifesto, the traditional traceability approaches must be adapted to fit the needs of stakeholders in these environments (Espinoza & Garbajosa, 2011).

As such, next to the typical automation focus which will be helpful for both traditional and agile environments, several studies focus on reducing the overhead of software traceability by focusing on approaches such as just in time traceability (JITT) that reduce the effort of trace creation and are more flexible and more appropriate for agile environments (Cleland-Huang, 2012). In addition, basic traceability as shown in figure 13 in agile environments is typically achieved between acceptance tests and user stories because of the focus on code and that this kind of traceability is easily obtainable and present in typical requirements management tools.

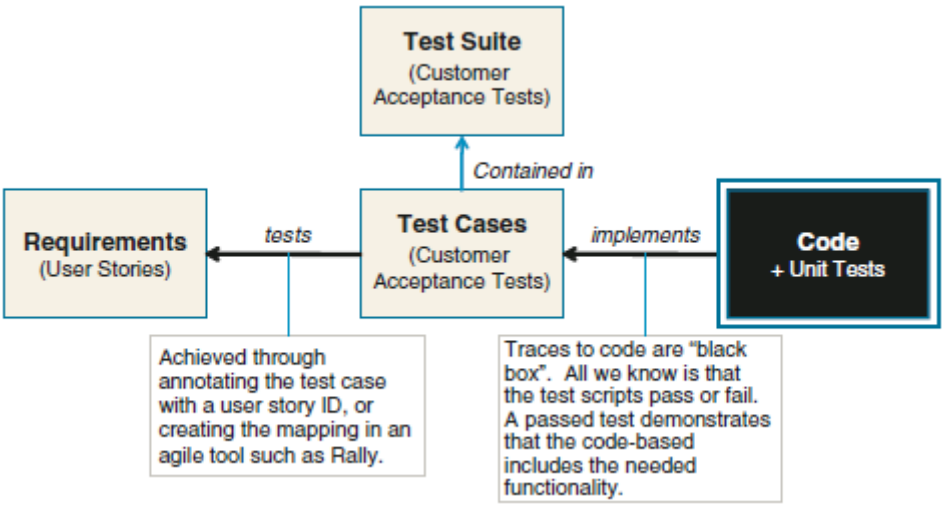


Figure 13: A basic traceability information model for basic agile projects (Cleland-huang, 2012).

Because of the extra overhead imposed by traditional traceability practices, it is found a decade ago that Agile practitioners generally perceive the notion of traceability in a negative light and that they perceive it as a heavy-weight and burdensome activity which returns little value to their projects, which is similar to the observations made several decades ago regarding the adoption and use of software traceability (Cleland-Huang, 2006, 2012). However, as per definition of software traceability adopted in section 3.1, and the observations according to the studies described in section 3.4, a possible explanation for this negative view could be due to the lack of awareness and guidance on what software traceability is and that some of the practitioners might think of the more traditional heavy weight approaches when the concept itself is brought up.

There have been efforts by the software traceability community to focus on software traceability in Agile SDLCs and there are several case based studies that focus on implementation of software traceability in agile environments (Furtado & Zisman, 2016). It is interesting to see that in some of these studies practitioners mention that traceability is important and that they can benefit from it, however, that there is a lack of guidance on traceability in agile environments (Alsalemi & Yeoh, 2016). However, besides the studies that focus on approaches, methods, and techniques developed for specific cases, there seems to be a lack of empirical survey studies that focus on the perspective of practitioners in agile environments. Even though some studies do contain samples of Agile practitioners, these kinds of studies are still in fewer numbers and are still a need for the software traceability community.

3.6 Benefits and costs of software and requirements traceability

As mentioned before, software traceability is typically a mandate in highly regulated environments, in particular in critical safety projects, by standards such as DO-178C (RTCA/EUROCAE 2011) and ISO 26262 (ISO 2011). However, besides the mandate from these standards, software traceability can also bring many benefits to the table. Researchers of software traceability mention benefits such as easier program comprehension, support for software maintenance, conducting impact analysis, ensuring enough test coverage, tracking project process, increased collaboration and many more that can come from a well-managed traceability process.

According to (Cleland-Huang et al., 2012), the value and benefits of software traceability lies in the traceability related tasks that it can enable such as impact analysis or being able to trace the rationale behind artefacts. In addition to the tasks that it can enable, traceability can also improve the

development processes by providing process related benefits such as increased time to delivery, increased transparency, and increased reliability.

However, little evidence exists and many of these benefits are difficult to measure because of the benefit problem described in section 3.4 and there are not enough experimental studies showing the benefits in a controlled environment.

Notable studies that do provide evidence regarding that benefits of traceability are for example the study of Mäder and Egyed, (2015), in which they conducted a controlled experiment with 71 subjects reperforming real maintenance tasks on development projects. Their findings showed that subjects with traceability performed on average 24% faster on a given task and created on average 50% more correct solutions, suggesting that traceability not only saves effort but can profoundly improve software maintenance quality.

Another study, by Rempel and Mader, (2017), where they focused on four main requirements implementation supporting activities that utilized traceability in 24 medium to large-scale open source projects, showed that more complete traceability decreases the expected defect rain the developed software. And that the strong impact of traceability completeness on the defect rate suggest that traceability is of great practical value for any kind of software development project, even if traceability not mandated by a standard or regulation.

Furthermore, in the study of Bouillon et al., (2013) which was briefly described in section 3.4, they systematically derived usage scenarios of requirements traceability which is shown in table 6.

1	**Requirements Engineering and Management**
a	Finding origin and rationale of requirements, i.e., pre-requirements traceability to regulatory and other source of a requirement
b	Refinement and detailing requirements
c	Documenting a requirement's history, i.e., to be able to trace to previous versions of a requirement in order to find out about changes
d	Identifying stakeholders for the ongoing development of the requirements
e	Quality- and maturity-analysis of requirements
f	Impact analysis, which other stakeholders are important by a change to a requirement
2	**Project Management**
a	Tracking the state of requirement or task implementation in detail
b	Initial Release planning
c	Progress assessment on project or subproject level for getting an overview of already implemented requirements
d	Task assignment to stakeholders, e.g., assignment of a requirement to a developer for implementation
e	Notification of stakeholders about changes, e.g., after a change to a requirement all owners of dependent artifacts are automatically informed
f	Adjusting project and release plan, e.g., in case of time limit exceeding
3	**Compliance Demonstration**
a	Analyzing requirements coverage in source code, e.g., for the customer
b	Traceability documentation for certification purposes
c	Justification of all written code based on specification for certification purposes
4	**Design and Implementation**

a	Navigation between specification, design, test, and code via traces
b	Navigation within artifacts of the same type, e.g., within source code
c	Design assessment based on traceability metrics, e.g., to find components that contain too much functionality and should be split
d	Understanding of software artifacts, e.g., project familiarization of development team members
5	**Testing**
a	Development of test cases based on requirements
b	Defect location within the source code for failed test cases
c	Discovering regression tests to be executed after code change
d	Test coverage analysis of specification and code
e	Stakeholder identification for understanding behavior and solving complicated problems
6	**Maintenance and Evolution**
a	Change impact analysis to determine artifacts impacted by a feature extension
b	Change effort estimation for feature extensions
c	Feature location and support during change implementation via use of traces
d	Reuse of specification and code components, e.g., a feature with all its implementation
e	Knowledge transfer to the maintenance team, e.g., in cases where a team performs maintenance that does not include any of to original team members

Table 6: User scenario catalog by Bouillon et al., (2013)

In addition to creating this initial catalog of user scenarios for requirements traceability they also performed a survey on how frequently these user scenarios were used in practice.
Their results showed that almost all scenarios were used and that the most frequently used scenarios were; finding origin and rationale of requirements, documenting a requirements history, and tracking requirement or task implementation state.

It is important to note however, that even though the study of Bouillon et al., (2013) focused on requirements traceability and thus requirements in particular. Many of these scenarios indicate to have similarities with the benefits mentioned by software traceability researchers. This is because, per definition, software traceability is an extension of the definition of requirements traceability as described in the introduction. Therefore, it is not a surprise that some of these user scenario benefits are also mentioned in software traceability research, considering that often the terms software traceability and requirements traceability are used interchangeably which adds extra confusion and intensifies the complexity of the software and requirements traceability research areas (Cleland-Huang et al., 2012). However, we assume that the benefits of software traceability, just like its definition, extends the benefits of these scenarios to not just the requirements but also other related development artifacts.

In practice, however, many of the described benefits are rarely realized because traceability in practice is often performed in an ad-hoc fashion (Cleland-Huang et al., 2014). Not only are the benefits rarely realized, practitioners also seem to struggle with conveying the benefits to their peers as described in section 3.4.

In addition, to having difficulty in conveying the benefits, it is more clearly known what the costs of traceability are. Much of the costs of traceability comes from the effort needed to create and maintain trace links, but also costs for education and tooling which ties back into the problems found in practice

that were described in section 3.4. e.g. cost-benefit problem and barriers of traceability adoption. These costs include financial costs but also cost in effort and time.

As such, even though there are many benefits to be expected from software traceability, actual evidence is scarce while the evidence of costs is clearer and more apparent. This intensifies the varying effort put in managing a traceability process in practice and is still in need for more attention.

3.7 Summary literature study

As described in section 2.1.1, the literature review serves to establish the knowledge for the fundamentals of software traceability, the current practice of research regarding software traceability, to refine and scope the initial objectives of this study, and to inspire and serve as basis for the contents of the questionnaire.

Based on this review, it became clear that there are still many issues regarding software traceability in practice and that there is a need for more contemporary evidence and data regarding stakeholders and users' perceived value of software traceability in both traditional and agile environments. In addition, we perceived an imbalance in research between these two environments.

Because of the different principles behind agile development compared to traditional development, some studies indicate that the value of software traceability can be perceived negatively, while on the other hand some findings of previous empirical studies seem to indicate that there is also a positive view on traceability from the agile practitioner's perspective. This prompts us to the following research question:

RQ4	*Is the perceived value of software traceability affected by the type of software development life cycle?"*

In addition to the imbalance between agile and non-agile environments regarding software traceability research, much of the research indicate to also be focused on safety-critical projects (Cleland-Huang et al., 2014). This is no surprise considering that traceability is mandated in these projects by standards and government regulations. However, even though it is mandated, it does not necessarily mean that traceability is valued among the practitioners in this environment. It is instead, possible that practitioners in these environments can see traceability in a negative light because of the mandate and see it as extra work. Even without the mandate, which is more common in non-critical safety projects, traceability could prove to be of practical value as shown in several of the described studies (Mäder & Egyed, 2015; Rempel & Mader, 2017). As such, we formulated the following research question:

RQ5	*Is the perceived value of software traceability affected by the type of software project?" (type = critical/non-critical)*

The reasons of investigating perceived value was described in the introductory sections of this study and cost benefit section (sections 1.3 & 3.6). In summary, even though there are many expected benefits, the evidence for it is scarce while on the other hand the costs for investing in software traceability are clearer. This, together with the lack of empirical evidence from a user perspective as described in section 3.4, prompts us to question how the practitioners perceive the value in terms of their needs in practice and the differences between the perceived value between safety critical, non-safety critical, Agile, and Traditional projects and environments. To be able to know how what the differences are, we first need to know how perceived value can be measured, which prompts us to the following question:

RQ3	How can perceived value be measured?

The knowledge gathered from the literature review of research questions 1 and 2, and the results of research question 3 serves as input for the design of the questionnaire. The design and the contents and hypotheses building of the questionnaire are described in the following section 4.

3.8 How can perceived value be measured?

In the section 1.3 we briefly touched on the topic of why the investigation is about perceived value, and that the perspective that we chose to view perceived value from is via the means-end theory perspective which focuses on the needs of traceability in software development projects.

Perceived value is an extensive researched concept in the marketing and consumer theory (Fernandez & Bonillo, 2007). There are multiple research streams and different ways of viewing the concept, but perceived value is usually viewed from two common perspectives: the unidimensional perspective and the multi-dimensional perspective.

One of the simpler ways to look at perceived value is in terms of its perceived benefits and perceived costs in terms of price and or functionality. This kind of view is usually adopted in price-based studies where the worth of the product or service is translated to a certain price tag. In addition, it is also adopted in means-end theory studies where the worth is based on how a certain product or service can help in achieving a goal, means, or end that the consumer has in mind. The price-based theory and means-end theory together view perceived value from a uni-dimensional perspective, in which according to Fernandez & Bonillo, (2007):

> *" perceived value is viewed as a single overall concept that can be measured by a self-reported item, or set of items, that evaluates the consumer's perception of value"*

In addition, the uni-dimensional perspective does include the possibility that the perceived value might be produced by the effects of multiple antecedents, but it does not include the view that value is an aggregate concept formed from several other complex components and concepts (Fernandez & Bonillo, 2007).

In contrast to the uni-dimensional view, the other view perceives the construct of perceived value as (Fernandez & Bonillo, 2007):

> *"a multi-dimensional construct that consists of several interrelated attributes or dimensions that form a holistic representation of a complex phenomenon"*

From this perspective, perceived value is seen as a construct that is made up of other and more complex underlying constructs. It can for example be based on a 'value hierarchy model' in which the perceived value can be measured based on consumption goals, consequences, attributes, and desired value and received value (Fernandez & Bonillo, 2007). On an abstract level, this view will contain more concepts and constructs that make up perceived value than just for example the benefits and costs. Another view regarding the multi-dimensional approach is looking at perceived value from both a utilitarian and hedonic view. According to Fernandez & Bonillo, (2007):

- the utilitarian value view touches upon the more instrumental, task-related, rational, functional, cognitive, and a means to an end sort of way; and

- the hedonic value view takes a more entertainment, emotional, non-instrumental, experiential, and affective perspective.

Besides these two different multi-dimensional approaches, there are a variety of other ways on how perceived value could be viewed as well and be the aggregation of a variety of concepts and constructs. The strength of this perspective is that it adds more richness to how perceived value can be viewed. However, because of the introduction of extra concepts and constructs, it also becomes more complex. An overview of the pros and cons of each view can be seen in table 7, which is reproduced based on the work of Fernandez & Bonillo, (2007).

Uni-dimensional nature	Multi-dimensional nature
- Roots in economic theory and cognitive psychology	- Roots in consumer-behavior psychology
- Utilitarian and economic conception	- Behavioral conception
- Cognitive approach	- Cognitive-affective approach
- Simplicity	- Richness and complexity
- Knowledge of how value is evaluated	- Specific direction on how to improve value
- Lack of agreement regarding the antecedents of value	- Lack of agreement regarding the components of value
- Confusion about the relationship among the antecedents	- Confusion about the relationship among the components
- Direct observation of value	- Observation of value through its components
- Widely embraced in the literature	- Hardly embraced in the literature

Table 7: Views on the nature of perceived value

The view that we have chosen to adopt during this study regarding perceived value is more closely related to the means-end theory perspective and the definition of perceived value according to Cambridge as described in section 1.3. This means that when perceived value is mentioned in this study, it refers to the simpler construct of perceived value which is made up based on the needs. The reason for this choice is that this view is enables us to be able to touch upon a variety of topics compared to the more widely accepted price-based perspective that mainly focuses on economic or financial benefits and costs. However, it is important to understand and be aware that this is not the only possible way to view and define perceived value.

Even though perceived value is difficult to measure as explained, in this study we try to get an indication of the perceived value based on what is needed of traceability and their ranking based on priority and importance. With the underlying assumption that the more something is needed, to more it is perceived as adding value. Logically speaking however, needs usually arise based on the perceived current situation in which people find themselves and what they want to improve based on their current situation and what they want to achieve. Because of this, we will focus on both the needs of practitioners regarding software traceability but also their perceived current situation regarding traceability. In other words, we will be measuring perceived value as an abstract construct in terms of the current situation of practitioners and in terms of their needs.

4. Electronic questionnaire design

To answer research questions 4 and 5 from a quantitative perspective a questionnaire is designed and used to collect data. The following sections detail the design of the questionnaire and the rationale behind the decisions.

4.1 Questionnaire design GQM

For the questionnaire we started with designing a high level Goal Question Metric (GQM) like tree, as seen in figure 14, which shows the goals that are intended to be achieved, the related questions, and the related hypotheses (Basili et al., 1994; van Solingen Rini et al., 2002). The difference between this figure and a traditional GQM is that a traditional GQM usually includes metrics, which we changed to hypotheses for our case since this is still on a higher level of abstraction.

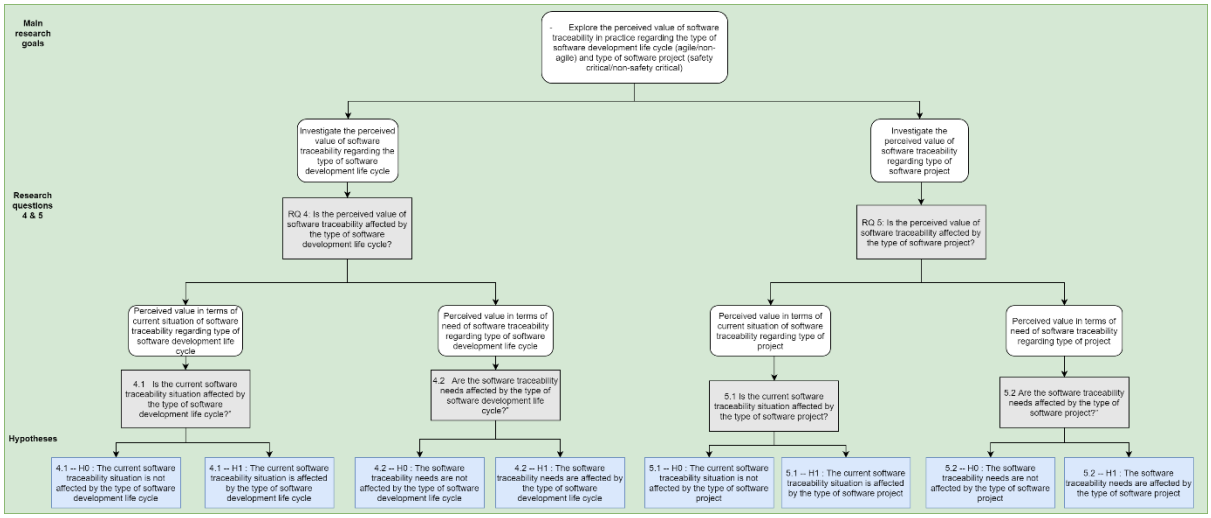


Figure 14: Hypotheses GQM

In figure 14, the model starts off with the main research goal of research questions 4 and 5, which were derived from the literature study in section 3. The main overarching goal of these questions is to explore the perceived value of software traceability in practice regarding the software development method (agile/non-agile) and the type of software project (safety critical/non-safety critical). With each of the research questions, 4 and 5, focusing on either the paradigm behind the development method or the type of software project respectively.

As described in section 3.8, we will be measuring perceived value in terms of their current situation and their needs. As such, as shown in figure 14, both research question 4 and 5 have two additional sub research questions which focuses on the current situation on one hand, which are research questions 4.1 and 5.1, and focuses on the needs on the other hand, which are research questions 4.2 and 5.2.

For each of these sub research questions, we have added related quantitative hypotheses, which are the null and alternative hypotheses. These hypotheses can then be tested in a quantitative fashion to see if there are differences in the perceived value, or if perceived value is affected differently, based on e.g. the different development methods used or the type of project regarding the current situation and needs.

With a better overview of the decomposition of the goals of research question 4 and 5 and their related hypotheses, the next artifact that we designed is a GQM like model for perceived value itself, as shown in figure 15.

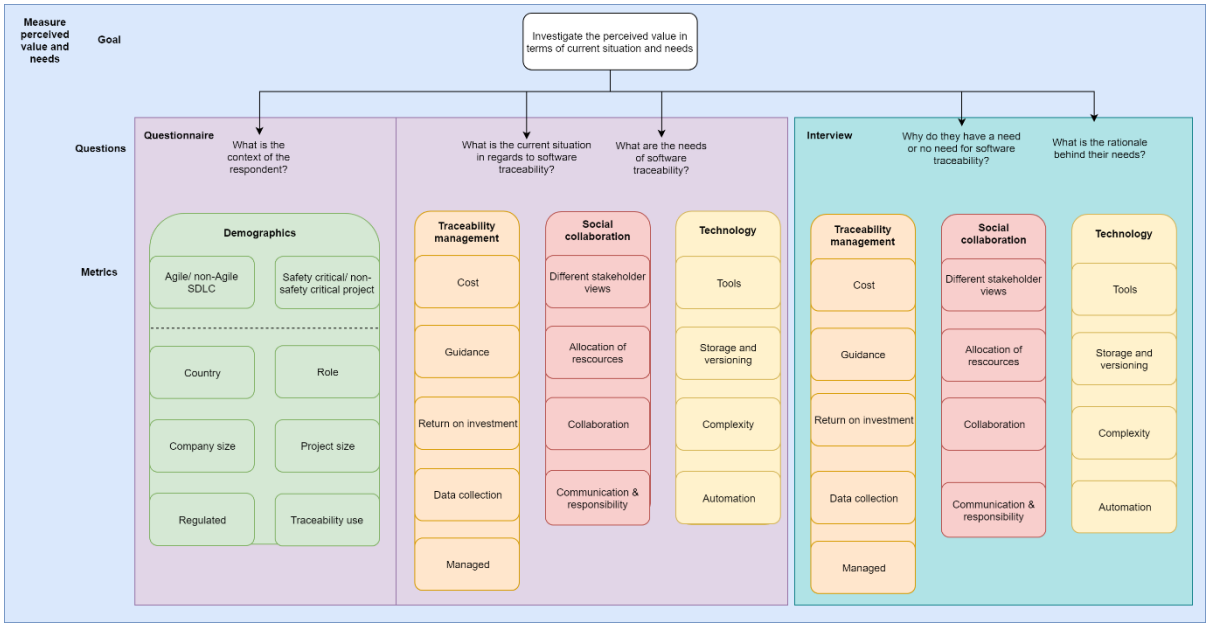


Figure 15: Perceived value GQM

For measuring perceived value in terms of current situation and needs, we used the categories of Regan et al., (2012) as the main reference framework and structure, which was described in section 3, for our questions and statements. The rationale behind this is that their reference framework, which is about the barriers of software traceability in practice, is based on their systematic literature review of the available empirical studies in regard to traceability in practice till 2012, which also covers most of the same literature and topics and or issues in practice that were discussed in section 3. Because of this, we think that the reference framework they made can be suitably adapted and used for this study to serve as a solid theoretical foundation for our questionnaire, in which the categories have already been tested and validated by another group of researchers. However, since they focus on issues, and we focus on the needs and current situation, we had to make a few small changes and adaptations to their framework. One of the changes was changing the names of the categories to be more abstract and general and to be not solely focused on issues. In addition, we omitted some of the categories and substituted a few other categories which were derived from other studies to suit the nature of our statements. Furthermore, the main difference is that each of the subcategories have their own statements based on the problems perceived in practice which are then proposed to the participants to understand their perceptions, perceived problems, but also their needs on them, compared to the potential solutions that Regan et al., (2012) proposed.

As seen in figure 15, both the current situation and needs will be measured via three main pillars/categories, which are dubbed as: “Traceability management”, “Social collaboration”, and “Technology”. Each of these pillars have their subcategories, e.g. “Traceability management” has Costs, Guidance, ROI, Data collection, Managed. Each of these subcategories in turn have their own statements that is related to their subcategory. The three pillars and the subcategories serve as the main structure of our questionnaire.

For the questionnaire we set up the following main requirements as to what the designed questionnaire should satisfy:

- Short, max 15 min.
- Lightweight, low effort to fill in.
- Anonymous

Because we wanted the questionnaire to be lightweight, to further balance the load and not make a too long of a questionnaire, we decided the questionnaire to mainly focus on the 'what' type of questions (Peterson, 2014). Typically, it is important to not only know the 'what', but also the 'why' or 'how'. In our case, the questionnaire focuses on the 'what', e.g. What is the current situation in regards to software traceability, while the interview will focus more on the 'why', e.g. why they have a need for something and the rationale behind it. The reason for this, is that it is difficult to ask 'why' type questions in questionnaire since you will typically end with open ended questions. These questions usually take a longer time to fill in and more effort and might deter participants from filling in the survey. As such, we focus in the questionnaire on the 'what' type questions and try to keep it short and general, while in the interviews we will dive into further detail and the rationale behind the answers of the participants that have filled in the questionnaire.

There is one exception, however. The questionnaire does ask the rationale in the case the answers of the current situation and needs seem to contradict each other. E.g., if they strongly agree with that costs is the main inhibitor but when they also answer that there is no need to decrease the costs. This logic is implemented in the questionnaire, and the appearance of these question will be dependent on the answers of the participant.

Finally, both the questionnaire and interview will be complementary and follow the same structure.

4.2 Statement formulation

Regarding the statements, for each subcategory of the pillar we brainstormed multiple statements per subcategory over multiple sessions together. Because one of the main requirements is to keep the questionnaire short, we eventually selected one statement per subcategory. The final list of statements that were selected, revised, refined, and validated per subcategory is seen in table 8. Each of the statements were derived either based on the literature that was discussed in section 3 or based on our own curiosity and the discussions during the meetings. A simple literature statement mapping is included in appendix G**.**

Category	#	Subcategory	Type	#	Statement
Management	1	Cost	Current situation	1.1	Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices
			Need	1.2	There is a need to reduce the costs of traceability
	2	Guidance	Current situation	2.1	There is insufficient guidance within the company on how to establish traceability
			Need	2.2	There is a need for more guidance in the company regarding traceability
	3	Return on Investment	Current situation	3.1	Traceability costs outweigh the expected benefits
			Need	3.2	There is a need to have a more clear overview of the costs and benefits regarding traceability
	4	Data collection	Current situation	4.1	It is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability
			Need	4.2	There is a need to have easier access to information sources to be able to establish and maintain traceability
	5	Managed	Current situation	5.1	Traceability is mostly performed in an ad-hoc and non managed fashion
			Need	5.2	There is a need to perform traceability in a more managed fashion

Social collaboration	6	Different stakeholder viewpoints	Current situation	6.1	Traceability is of high importance for the software development process
			Need	6.2	There is a need to increase the awareness of the importance of traceability
	7	Allocation of resources	Current situation	7.1	The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability
			Need	7.2	There is a need for more staff, time, and resources to properly establish and maintain traceability
	8	Collaboration	Current situation	8.1	There is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability
			Need	8.2	There is a need for more collaboration between involved stakeholder teams regarding traceability
	9	Communication & responsibility	Current situation	9.1	It is unclear which roles are responsible for traceability
			Need	9.2	There is a need for more communication about who are responsible for traceability
Technical	10	Tools	Current situation	10.1	Traceability tools do not satisfy our traceability needs
			Need	10.2	There is a need for better traceability tools
	11	Storage and versioning	Current situation	11.1	It is difficult to establish traceability because development artifacts are stored in multiple locations/repositories
			Need	11.2	There is a need for a more centralized development artifact repository to establish traceability more easily
	12	Complexity	Current situation	12.1	The tools used for traceability are too complex to use effectively and to integrate with our existing tools
			Need	12.2	There is a need for less complex traceability tools and easier integration with our existing tools
	13	Automation	Current situation	13.1	Traceability is mostly performed manually
			Need	13.2	There is a need for more traceability automation

Table 8: Final selected set of statements

Regarding the possible answers for the statements, we decided to go with Likert scale choices in which the statements regarding the current situation (every x.1 statement) can be answered on a scale from:

- Strongly disagree
- Somewhat disagree
- Somewhat agree
- Strongly agree
- N/A.

The reason for using a 4-point scale with an extra N/A opposed to the typical 5 or 7 Likert scale possibilities is to make sure that the participants think about the statements instead of answering 'neutral' or similar kinds of answers. The N/A option is supposed to be only chosen if the statement is not applicable or if the participant does not want to answer that specific statement.

Furthermore, for the need type statements (every x.2 statement), we based our scale loosely on the scale of the MoSCoW prioritization technique which is commonly used for prioritizing requirements in software development projects (Miranda, 2011; Waters, 2009). This MoSCoW technique prioritizes the requirements in the following priorities:

- Must have
- Should have
- Could have
- Won't have

Since the official MoSCoW is focused on requirements and deals with specific timeboxes, for example 'Must have' denotes that it is critical to finish a specific requirement in the current timebox/period of development, we adapted the MoSCoW scales in our case to make more sense with the statements considering we are not necessarily dealing with timeboxes:

- Must have
    - Critical need
- Should have
    - Important need but not critical
- Nice to have
    - Desirable need but not necessary
- Not needed
    - Not of importance
- N/A
    - If non applicable or not willing to answer

Furthermore, with our custom Likert scale, the more a participant prioritizes a certain need, the more important, or the more it is perceived of value to the participant.

Next to the statements, we included several demographic questions as show in table 9.

Demographic	Country	1	In which country do you currently reside?
	Company size	2	What is the size of your company?
	Role	3	Which role are you typically assigned to during software development projects?
	Project size	4	On average, how many people are involved in the software development projects you participate in?
	Type of SDLC	5	Which software development paradigm is normally used in the projects you are involved with?
	Regulated	6	Would you characterize the industry you work in to be highly regulated?
	Type of project	7	Would you characterize your software as safety-critical?
	Traceability use	8	How often is software traceability performed in the projects that you are involved in?
		8.1	Your reason(s) for performing traceability

Table 9: Demographic questions

For our research questions, the main demographic variables that are of importance are the type of development method/software development cycle that is being practiced and the type of project in terms of safety critical vs non-safety critical.

Regarding the development method, instead of asking which specific method is being used, we asked the nature or in other words which paradigm is being followed. The participants can choose from:

- Agile, which includes SDLCs such as Scrum, Kanban, XP, and any other forms that follow the Agile paradigm in nature.
- Traditional, which includes SDLCs such as waterfall, V-model, and any other forms that follow the Traditional paradigm.
- Mixed, which includes custom SDLCs that combine the values of both Agile and Traditional paradigms.

Regarding the type of project, we ask if its safety critical or not regarding if the software and or system that is being developed in a project can have a huge impact on the safety or lives of people using it.

Next to the main demographic variables, we added a few complementary variables to be able to group our data and which were of interest for us. These demographic variables include:

- Company size, in which the answer scale is based on the size definition for small-medium enterprises (SME) as provided in the guidelines by the European Commission, in which the possible options are (European Commision, 2015):
    - Micro (1-9)
    - Small (10-49 people)
    - Medium (50-249 people)
    - Large (250+ people)

- Role, in which the typical development roles were added based on Agile development processes and Traditional development processes. Furthermore, the participants have the option to specify roles that are not shown in the initial set:
    - Product Owner
    - Project Manager
    - Software Developer
    - Software Tester
    - Software Architect
    - Requirements Analyst
    - Other

- Project Size, in which the participants can choose from the following sizes:
    - 1-9 people
    - 10-29 people
    - 30-89 people
    - 90+ people

    Regarding the project size, the project size can be interpreted in a variety of different ways such as meaning only the core team, or the core team plus other people who have helped with the project but are not part of the core team. In addition, it can vary as to which roles and who are part of the core team. To simplify this, we decided to add an extra explanation which details that in our case, project size is how many people, on average, are involved with the development project. In which the participants can think of the people that interact with

the software development artifacts such as requirements, architecture, code, test cases, user feedback, user manuals.

- Regulated, in which the participants can indicate if the industry they work in is highly regulated or not.

- Traceability use, in which the participants can indicate how often they use or perform traceability in their projects:
    - Always
    - Sometimes
    - Never

For the implementation of the questionnaire, we compared several questionnaire tools such as Google survey, SurveyMonkey, JOTFORM, and more. The tool we decided to use was JOTFORM, since it is free, simple to use and included all the features we needed. E.g. Google forms does not have enough features to include logic, and SurveyMonkey was not used because of licensing issues.

4.3 Questionnaire validation

The validation of the questionnaire was performed in multiple phases as shown in figure 16.

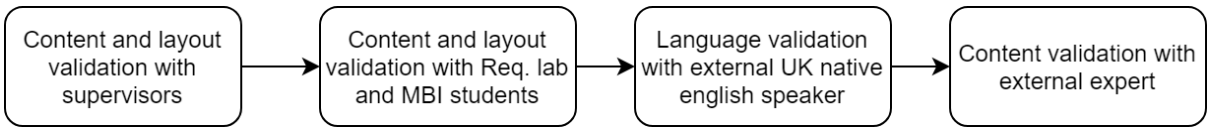


Figure 16: Validation phases

In the first phase, the content and the layout and the language went through the first round of validation in which the researchers together did an initial check until the content, layout, and language were satisfactory.

After the first phase, multiple MBI students were asked to participate in the content and layout validation by visiting the questionnaire and filling in a separate simple feedback form, which can be found in appendix H. Based on the feedback of the students, the initial questionnaire draft was refined and improved.

After that, the help of a native UK English speaker was asked to help in checking if there are issues with the language and if everything is understandable. The reason for including someone who speaks English natively is that the questionnaire will be available internationally, so it is important that the used terms are understandable and as simple as possible and grammatically correct.

After the validation with the students and native English speaker, the initial idea was to validate the contents, e.g. the questions and statements, with multiple experts and to see if the content makes sense for them or not, or if there are places where they had issues with answering, considering most of the statements were derived based on knowledge of the literature.

However, since we knew that we had a limited number of contacts that can participate, we wanted to keep the validation short and ask the rest of the experts or practitioners for participating in the actual survey instead asking them for the validation. So, in the end we opted to validate the content with one expert/practitioner instead.

Regarding the background of this expert, the expert in question has a range of experience mainly in practice and in multiple software development projects as either consultant but also developer.

The way in which the questionnaire was validated with the expert, was by means of a concurrent think aloud session. The main reason for choosing a concurrent think aloud session, opposed to using a retrospective think aloud session is so that the researcher could as questions and opinions on topics or statements that the expert would have missed otherwise or had questions about. This made it possible to gather more rich data by discussing certain points immediately, in comparison to retrospective think aloud sessions where the researcher asks questions in the end. In addition, think aloud sessions in general are suitable for validating designed artifacts regarding usability and to learn about the thought process when participants fill in our survey, and to pinpoint problem areas or design flaws.

	N	Content	Layout	Language
Supervisors	2	x	x	x
MBI students	6		**x**	x
Native English speaker	1		x	**x**
Expert	1	**x**	x	x

Table 10: validation mapping

Based on the validation of each phase and the refinements, the final version of the questionnaire was produced which is included in appendix B.

In addition, since the validation was not done with as many participants as we initially wanted, in the final questionnaire the participants are provided with the option to give feedback, be it on the questionnaire itself or the topic of software traceability. This so that the questionnaire can be refined for future iterations and so that potential threats of the current questionnaire can be spotted based on their feedback.

The final version of the questionnaire was open and distributed in the period between 19 June and 8 July. In the end, because of the low response rate, the distribution was extended to 31 July.
The questionnaire was mainly distributed via personal contacts, LinkedIn, and Reddit.

4.4 Intermezzo: Initial perceived threats

Before distributing the questionnaire, we were aware of several possible treats. The main threats are reliability of data, ambiguity and bias.

Reliability of the data, the survey is an anonymous online survey that is published on social media. This has implications to the reliability of the data since we cannot trace back the background of the participants and check if the participants were part of our target demographic. In addition, sources from Reddit could for example be less reliable compared to the sources from LinkedIn. Nonetheless, the reason for keeping it anonymous and putting it on social media is to collect as much information as we can and to reach as many people as possible. The reason for the anonymity is mainly because since the questionnaire deals with the opinions and experiences of people, the anonymity will give them more freedom to be as candid as they want with their answers without having to worry about the consequences. Especially considering, based on the literature, that there are many mixed feelings about the topic itself.

Ambiguity, even though we have tried to simplify the statements as much as possible and validated it with multiple students, a native English speaker, and an expert. It is still always possible for there to be ambiguity and every person can read and interpret a statement differently. We tried to reduce it by simplifying the statements as much as possible and being as concise as possible, however, this threat remains regardless. In addition, even though we also tried to simplify and make the statements as self-explanatory as possible, it is also of benefit on the other hand to know how the participants interpret the statements differently. Even though no specific rationales are asked in most of the cases, except for contradictory statements, we hope to shed light on how people interpret the statements differently through the interviews themselves and what their views and explanations of traceability are and the value behind it.

Bias, the way that the survey is designed is so that everyone of the target demographic can fill in the questionnaire. This makes it also possible that multiple people from the same company can fill in the questionnaire which might influence the results in the case these people are of same opinion because of their contextual factors. Nonetheless, the study is focuses on the practitioners and not necessarily on the companies. Therefore, we did not add unique identifiers for every possible company that could have responded and to keep it as anonymous as possible.

5. Overall questionnaire results & analysis

This section gives a brief overview of the data gathered from the questionnaire. It mainly describes descriptive results based on the overall participant population. Each subsection includes a relative frequency graph, a summary table, and a short description. Section 6 and 7 analyze the data in further detail based on the main subgroups of interest regarding the research questions as formulated in section 1. In addition, the descriptive results and the sample distribution and its impact on the analysis results are discussed in section 9.

For the questionnaire, there were in total 55 submissions. The data was preprocessed and cleaned to work with and ended up having 55 rows x 37 columns.

5.1 Demographic results

5.1.1 Country

The country graph as shown in figure 17 shows the distribution of how many of our participants come from a certain country.

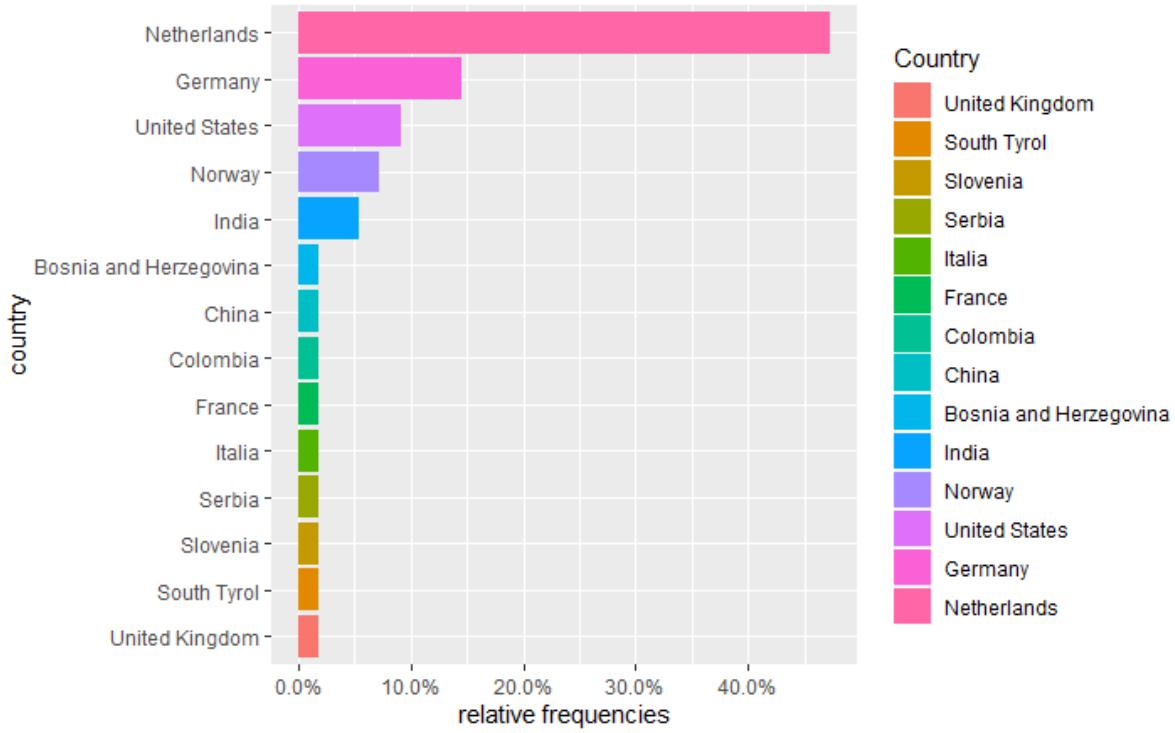


Figure 17: Relative frequencies of participant locations

#	Country	n	Percentage (%)
1	Netherlands	26	47.27
2	Germany	8	14.55
3	United States	5	9.09
4	Norway	4	7.27
5	India	3	5.45
6	Bosnia and Herzegovina	1	1.82
7	China	1	1.82

8	Colombia	1	1.82
9	France	1	1.82
10	Italia	1	1.82
11	Serbia	1	1.82
12	Slovenia	1	1.82
13	South Tyrol	1	1.82
14	United Kingdom	1	1.82

Table 11: Summary table of participant locations

As can be seen in figure 17 and table 11, 47.27% of our participants come from the Netherlands, followed by 14.55% from Germany, 9.09% from United States, 7.27% from Norway, 5.45% from India, and the rest of the countries have 1 participant each with 1.8%. In addition, as shown in figure 18, most of the participants are in Europe, followed by America, and finally Asia.

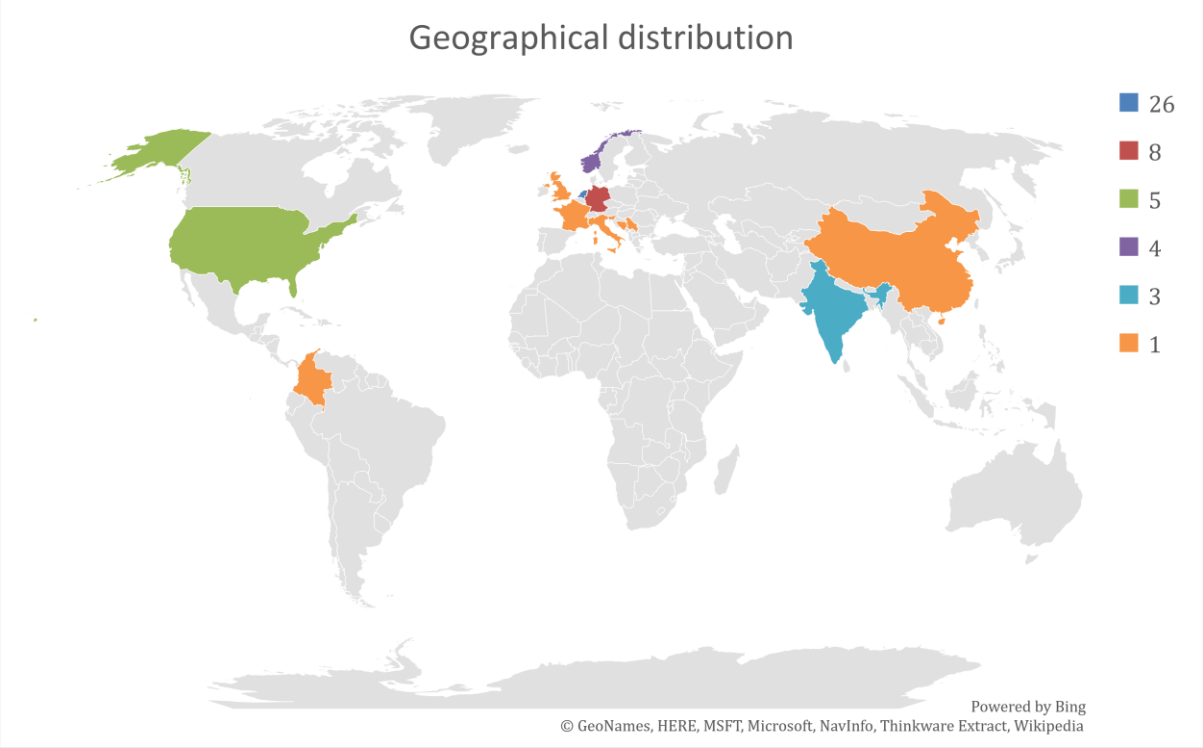


Figure 18: Geographical distribution

5.1.2 Company Size

Figure 19 and table 12, show the company sizes of the participants of the questionnaire. Most of the participants with 61.82% are from large companies with over 250+ people. This is followed by 16.36% of the participants from medium companies with between 50 and 249 people, 12.73% from small companies with 10-49 people, and finally 9.09% are from micro companies with between 1 and 9 people.

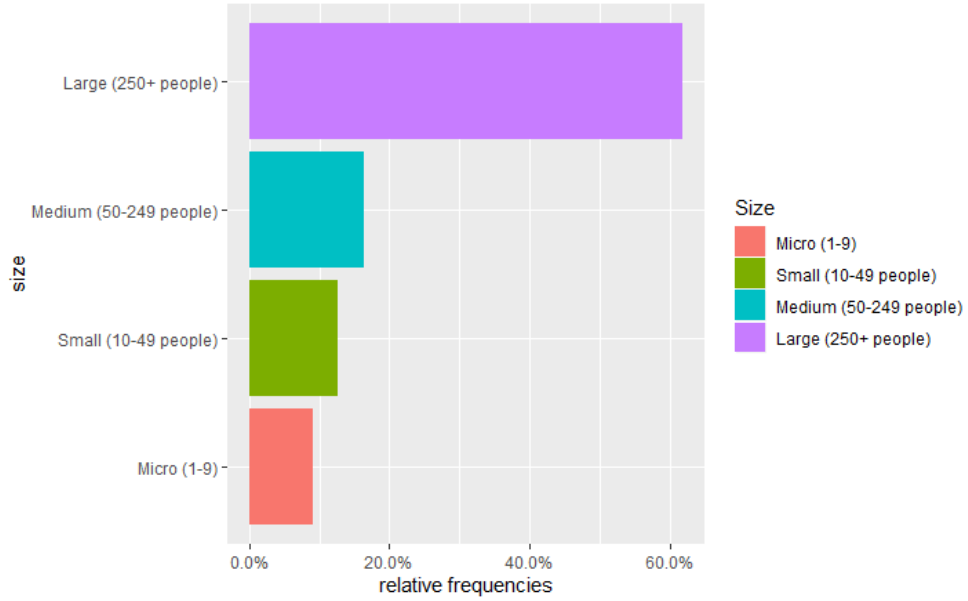


Figure 19: Relative frequencies of company sizes of participants

#	Size	n	Percentage (%)
1	Large (250+ people)	34	61.82
2	Medium (50-249 people)	9	16.36
3	Small (10-49 people)	7	12.73
4	Micro (1-9)	5	9.09

Table 12: Summary table of company sizes

5.1.3 Roles of the participants

As can be seen in figure 20 and table 13, 27.27% of the participants work as a Software Developer, followed by Product Owner and Project Manager and Requirements Analysts each with 14.55%. Furthermore, 10.91% work as a Software Architect, 5.45% as Software Tester, and the rest of the roles have one participant each with 1.82%.

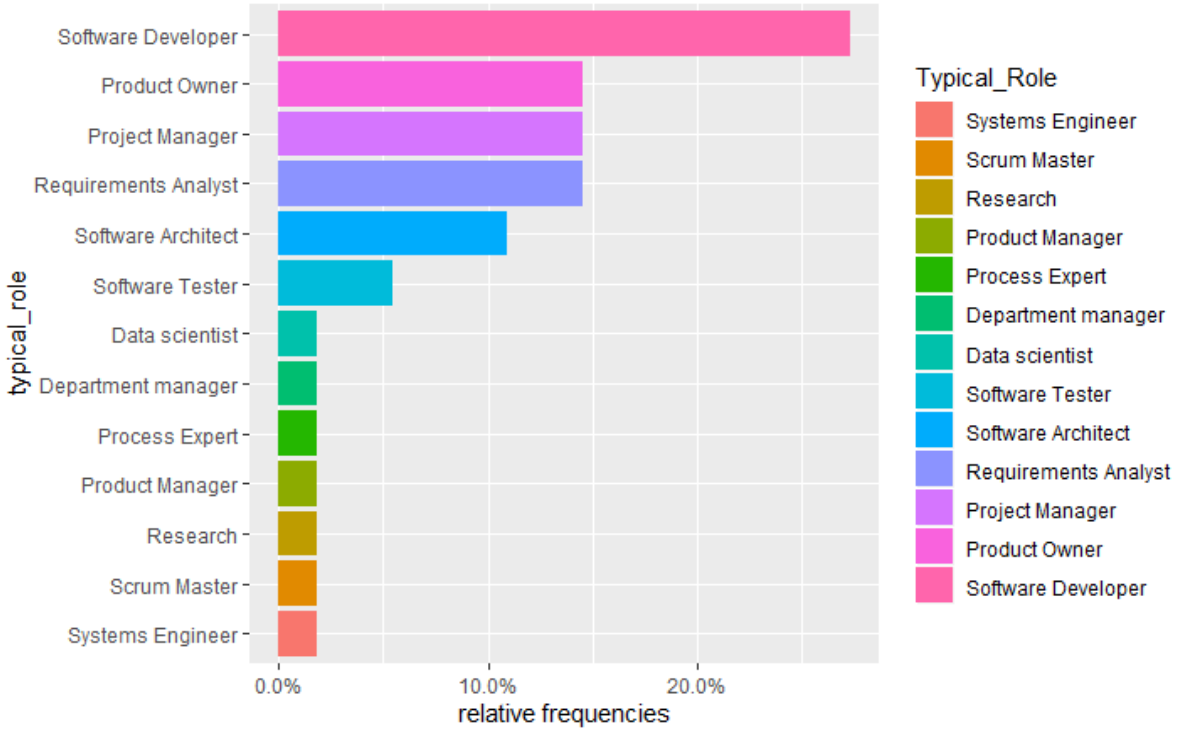


Figure 20: Relative frequencies of roles of participants

#	Typical_Role	n	Percentage (%)
1	Software Developer	15	27.27
2	Product Owner	8	14.55
3	Project Manager	8	14.55
4	Requirements Analyst	8	14.55
5	Software Architect	6	10.91
6	Software Tester	3	5.45
7	Data scientist	1	1.82
8	Department manager	1	1.82
9	Process Expert	1	1.82
10	Product Manager	1	1.82
11	Research	1	1.82
12	Scrum Master	1	1.82
13	Systems Engineer	1	1.82

Table 13: Summary table of roles

5.1.4 Project size

As shown in figure 21, and table 14, most projects in which our participants participate in have between 1 and 9 people with 41.82%, followed by 10-29 people with 38%, 18.18% with 90+ people and 1.82% with between 30 and 89 people.

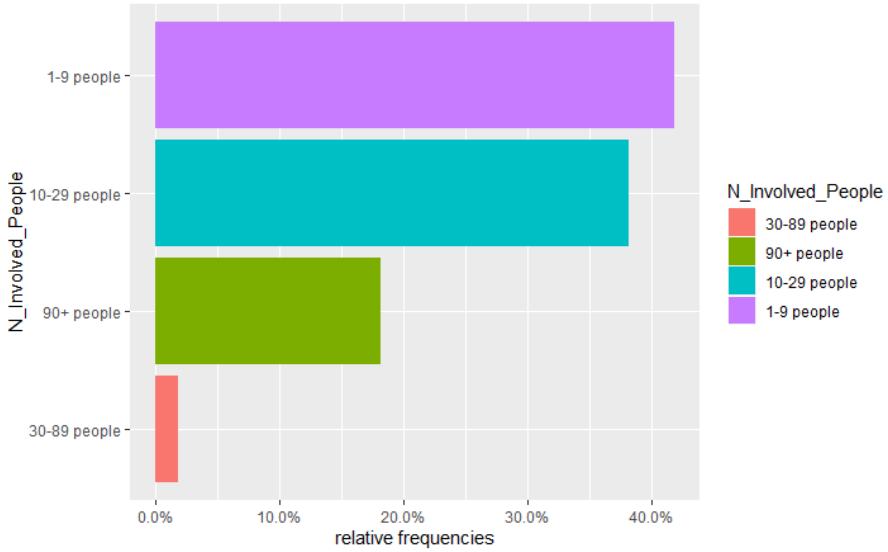


Figure 21: Relative frequencies of number of involved people in participants software projects

#	N_Involved_People	n	Percentage (%)
1	1-9 people	23	41.82
2	10-29 people	21	38.18
3	90+ people	10	18.18
4	30-89 people	1	1.82

Table 14: Summary table of number of involved people in participants software projects

5.1.6 Software development process: Paradigm

As can be seen in figure 22 and table 15, the percentage following an Agile paradigm is 67.27%, followed by a Mixed paradigm approach with 27.27%, and finally Traditional with 5.45%

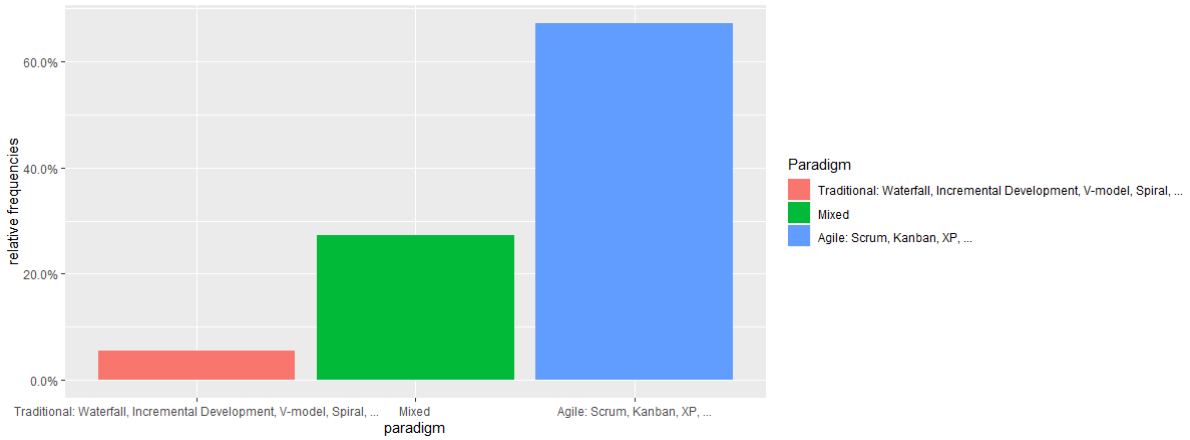


Figure 22: Relative frequencies of paradigms followed during projects

#	Paradigm	n	Percentage (%)
1	Agile	37	67.27
2	Mixed	15	27.27
3	Traditional	3	5.45

Table 15: Summary table of paradigms followed during projects

5.1.7 Industry regulation

As can be seen in figure 23 and table 16, 45.45% of our participants do not work in a highly regulated project, while 54.55% work in highly regulated projects.

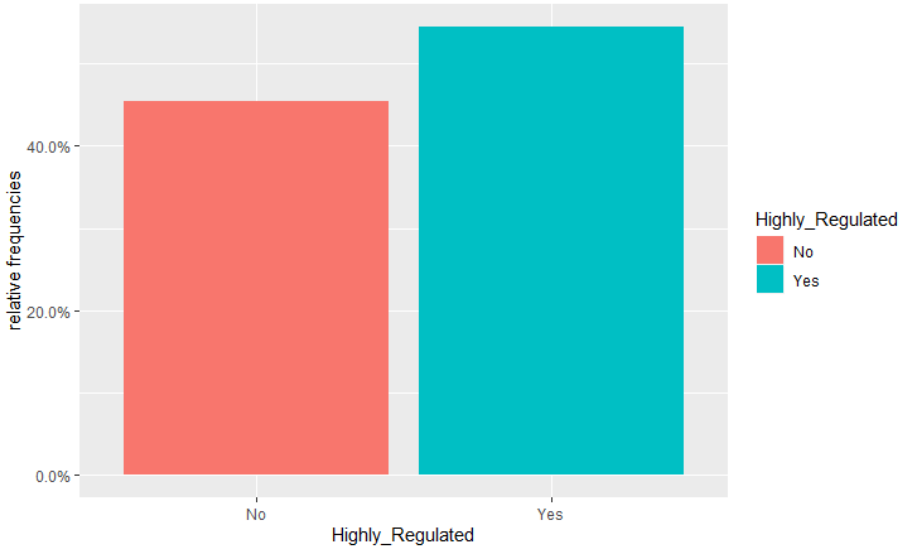


Figure 23: Relative frequencies of highly regulated projects

#	Highly_Regulated	n	Percentage (%)
1	No	25	45.45
2	Yes	30	54.55

Table 16: Summary table of regulated projects

5.1.7 Safety critical

As can be seen in figure 24 and table 17, 63.64% of our participants do not work in safety critical projects while 36.36% work in safety critical projects.

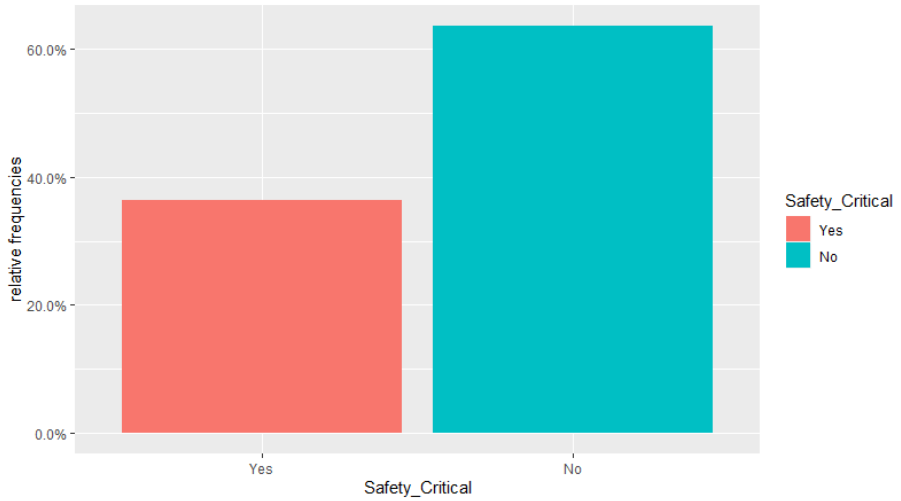


Figure 24: Relative frequencies of safety critical projects

#	Safety_Critical	n	Percentage (%)
1	No	35	63.64
2	Yes	20	36.36

Table 17: Summary table safety critical projects

5.1.8 Traceability usage

As can be seen in figure 25 and table 18, 50,91% of our participants sometimes use traceability in their projects, while 34.55% always use traceability and 14.55% never use traceability in their projects.

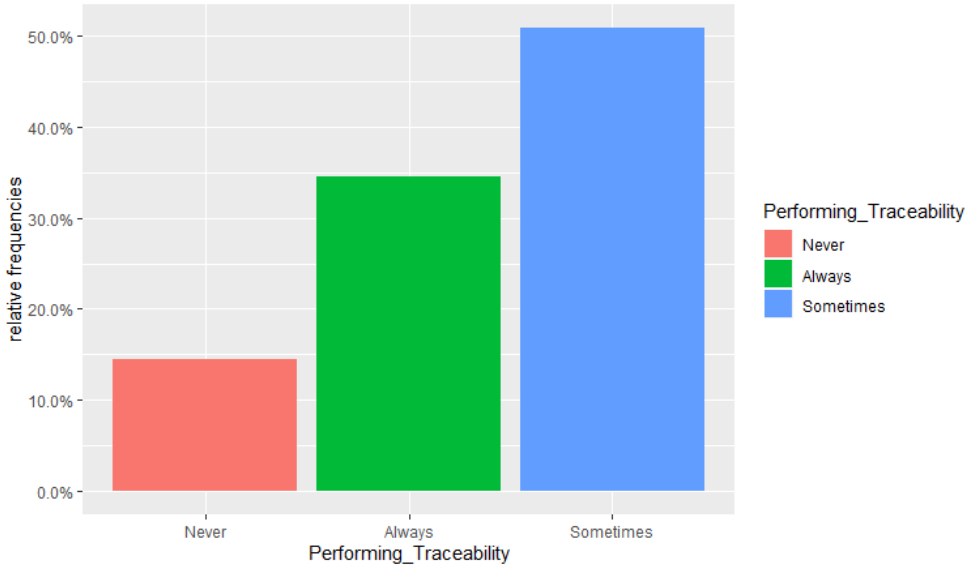


Figure 25: Relative frequencies of traceability use in projects

#	Performing_Traceability	n	Percentage (%)
1	Sometimes	28	50.91
2	Always	19	34.55
3	Never	8	14.55

Table 18: Summary table traceability usage

5.1.9 Traceability reason

As can be seen in figure 26 and table 19, 25,45% of our participants use traceability for the expected benefits, 14.55% did not use traceability and therefore did not indicate a reason why they are currently using it, 12.73% of the participants use traceability because of mandate, regulations, and the expected benefits, while the same percentage uses it just because of mandate and regulations. 9.09% of the participants use traceability because of mandate and regulations but also on requests of stakeholders and or customers, while the same percentage of the participants uses traceability for other reasons. Furthermore, 3.64% of the participants uses traceability because of mandate, on customer requests, and for the expected benefits, and the same percentage uses it just on requests and also for the expected benefits. The rest of the combinations include 1.82% of the participants each.

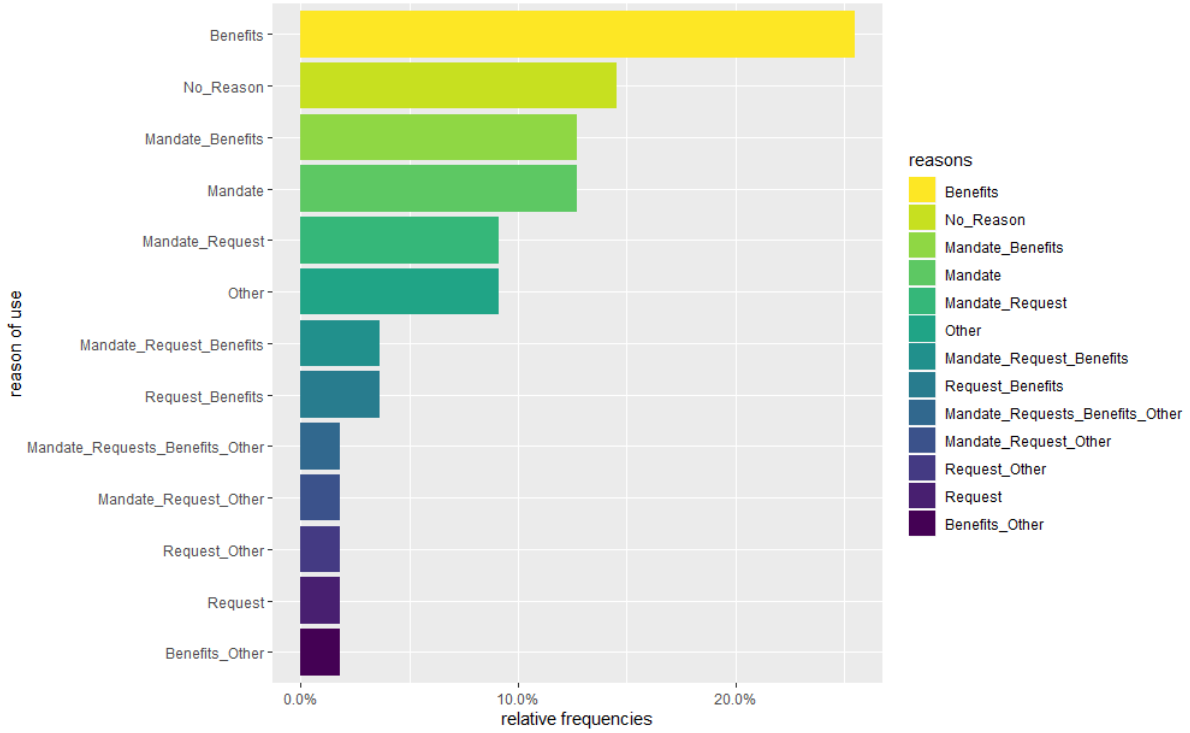


Figure 26: Relative frequencies of traceability reason of use

#	Reasons	n	Percentage (%)
1	Benefits	14	25.45
2	No_Reason	8	14.55
3	Mandate_Benefits	7	12.73
4	Mandate	7	12.73
5	Mandate_Request	5	9.09
6	Other	5	9.09
7	Mandate_Request_Benefits	2	3.64
8	Request_Benefits	2	3.64
9	Mandate_Requests_Benefits_Other	1	1.82
10	Mandate_Request_Other	1	1.82
11	Request_Other	1	1.82
12	Request	1	1.82
13	Benefits_Other	1	1.82

Table 19: Summary table reasons of traceability use

In addition, when participants included 'other' reasons, they mentioned:

- Accountability for request/change, investment tracking/reporting/analysis, relating information for later refinement information needs
- Lifecycle management
- Long-term knowledge about the requirements that are available in a product for maintenance and continuous product development
- Bug triage
- To improve their product

- Functional Safety
- CMMI and quality procedures
- To check whether there is a difference in business logic between different points in time, regulations and customer requests.

5.2 Descriptive Statement analyses

Each of the following sections describes the overall results of the statements and the overall sentiment of the participants via a divergent bar chart and a statistical summary table. In all the percentile calculations in the following sections the NAs are omitted, with the percentiles being based of the new totals after the NAs are subtracted of the total sample (N=55). The percentile calculations including NAs can be found in appendix D.

Divergent bar charts are good charts for visualizing sentiments of groups of participants and to get a quick overview of the trends and differences between groups based on how many of the participants chose a certain answer. In these charts, the size of the bars corresponds to the number of participants that chose a specific answer, in our case strongly disagree, somewhat disagree, somewhat agree, strongly agree, like a typical bar chart. However, in addition, the bars are centered around a zero line.

The zero line in our case for current situation statements separate a disagreeing sentiment on the left of the zero line, with an agreeing sentiment on the right of the zero line, including the percentiles. While the zero line for the needs statement separate sentiments between not needed, nice to have, and should have and must have.

5.2.1 Current statements

Figure 27 shows the divergent bar chart that shows the overall sentiment of all the participants regarding the current situation statements.

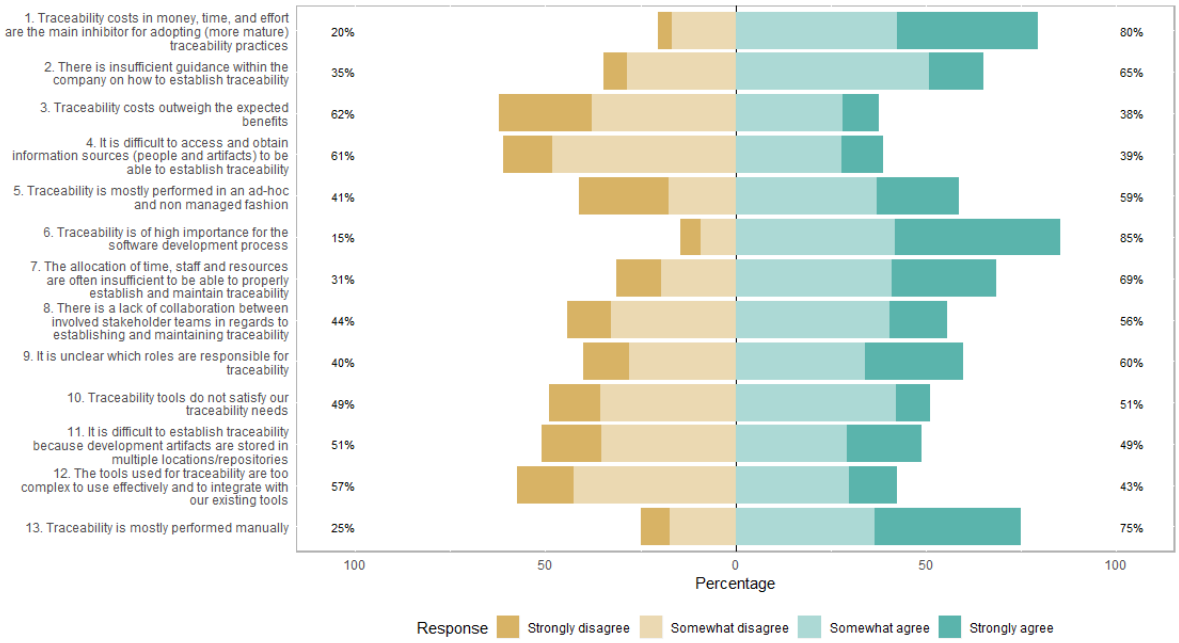


Figure 27: Divergent bar chart of overall sentiment regarding current situation statements.

As can be seen in figure 27, and the summary tables included in appendix D.1.3, D.1.4, and D.1.5, most of the participants:

1 Agree with statement 1 that traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices (disagree 20% < agree 80%, N = 54). With most of them somewhat agreeing, with 42.59% and with a numeric mean of 3.13 > 2.5[2].

2 Agree with statement 2 that there is insufficient guidance within the company on how to establish traceability (disagree 35% < agree 65%, N = 49). With most of them somewhat agreeing, with 51.02% and with a numeric mean of 2.73 > 2.5.

3 Disagree with statement 3 that traceability costs outweigh the expected benefits (disagree 62% >38%, N= 53). With most of them somewhat disagreeing with 37.74% and a numeric mean of 2.23 < 2.5.

4 Disagree that it is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability (disagree 61% > 39%, N = 54). With most of them somewhat agreeing with 48.15% and a numeric mean of 2.37 < 2.5.

5 Agree that traceability is mostly performed in an ad-hoc and non-managed fashion (disagree 41% < 59%, N = 51). With most of them somewhat agreeing with 37.25% and a numeric mean of 2.57

6 Agree that traceability is of high importance for the software development process (disagree 15% < 85% agree, N = 55). With most of them strongly agreeing with 43.64% and a numeric mean of 3.24 > 2.5.

7 Agree that the allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability (disagree 31% < agree 69%, N = 51). With most of them somewhat agreeing with 41.18% and a numeric mean of 2.84 > 2.5

8 Agree that there is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability (disagree 44% < agree 56%, N = 52). With most of them somewhat agreeing with 38.46% and a numeric mean of 2.65 > 2.5.

9 Agree that it is unclear which roles are responsible for traceability (disagree 40% < agree 60%, N = 50). With most of them somewhat agreeing with 34% and a numeric mean of 2.74.

10 Agree that traceability tools do not satisfy their needs (disagree 49% < agree 51%, N = 45). With most of them somewhat agreeing with 42.22% and a numeric mean of 2.47 < 2.5 indicating that the participants more strongly disagree then strongly.

11 Disagree that it is difficult to establish traceability because development artifacts are stored in multiple locations/repositories (disagree 51% > agree 49%, N = 51). With most of them somewhat disagreeing with 35.29% and a numeric mean of 2.53 > 2.5

[2] Likert scale responses range from strongly disagree, somewhat disagree, somewhat agree, to strongly agree. By changing these responses to their numeric variants, they correspond to 1-2-3-4 respectively. This results in that a mean of 2.5 is considered neutral, with means above indicating they agree more with the statement and means below indicating they disagree more with the statements.

12 Disagree that tools used for traceability are too complex to effectively use and to integrate with their existing tools (disagree 57% > agree 43%, N = 47). With most of them somewhat disagreeing with 42.55% and a numeric mean of 2.4

13 Agree that traceability is mostly performed manually (disagree 25% < agree 75%, N = 52). With most of them strongly agreeing with 38.46% and a numeric mean of 3.06 > 2.5

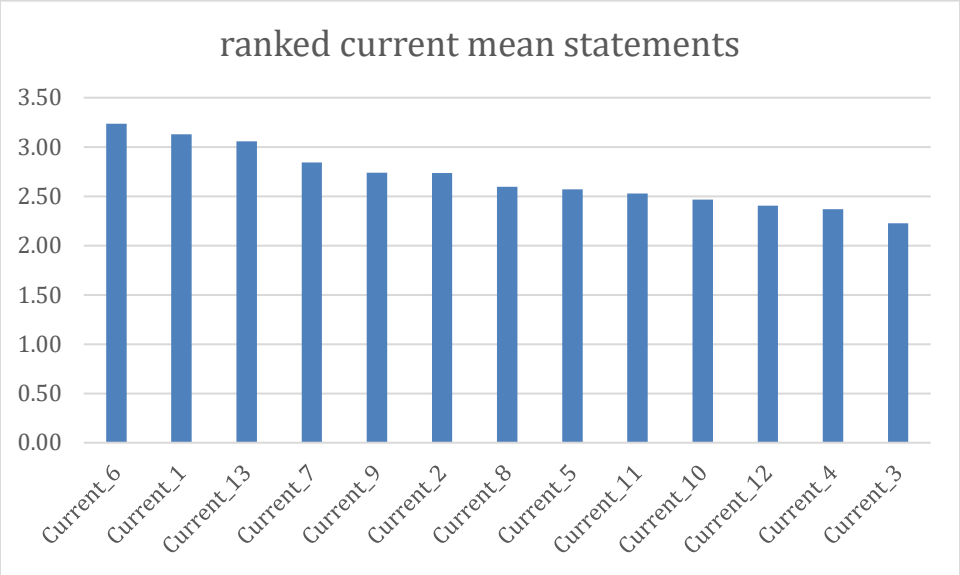


Figure 28: Perceived current situation statements ranked from strong agreement to strong disagreement based on mean.

By ranking the statements from high to low based on the numeric means, in which the higher values indicate a stronger agreement and lower values indicate a stronger disagreement, it is possible to see which statements the participants agree or disagree the most with, as seen in figure 28.

In regards to agreeing with a statement, the results show that based on the means the participants agree most with Current_6: *traceability is of high importance for the software development process*, Current_1: *traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices*, and Current_13: *traceability is mostly performed manually*.

In regards to disagreeing with a statement, the results show that based on the means the participants disagree most with Current_3: *traceability costs outweigh the expected benefits*, Current_4: *it is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability*, and Current_12: *tools used for traceability are too complex to effectively use and to integrate with their existing tools*.

5.2.2 Need statements

Figure 28 shows the divergent bar chart that shows the overall sentiment of all the participants regarding the perceived value of traceability in terms of their needs.

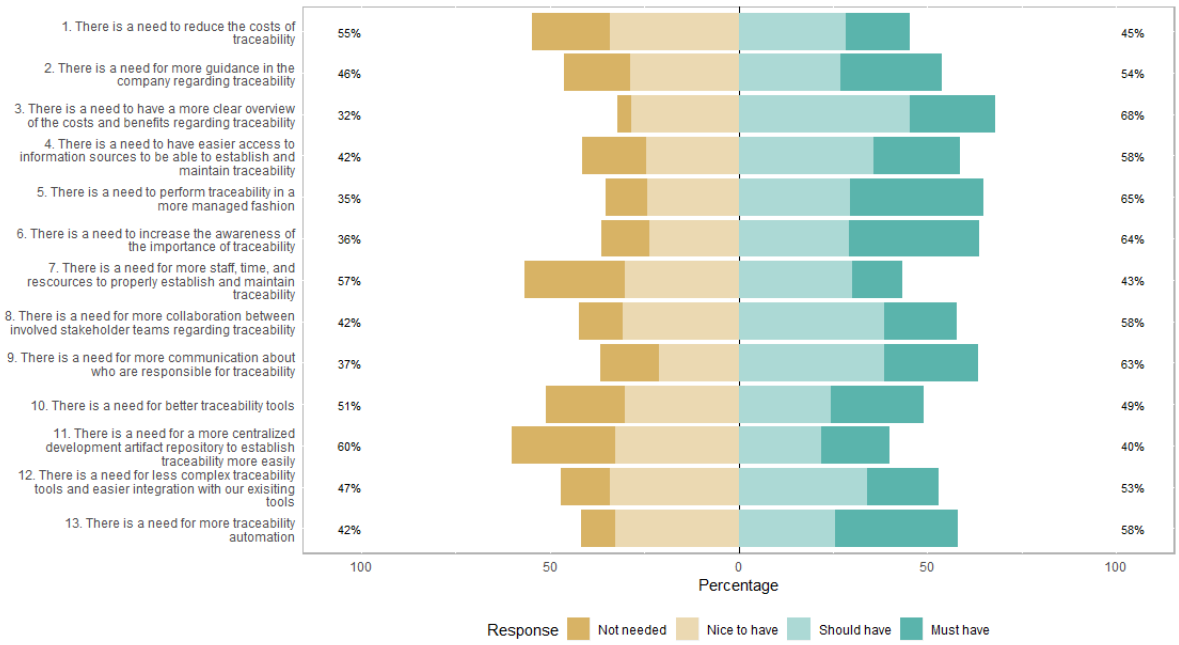


Figure 29: Divergent bar chart of overall sentiment regarding perceived value of software traceability needs.

As can be seen in figure 29, and the summary tables included in appendix D.2.3, D.2.4, and D.2.5, most of the participants:

1 Either not needed or nice to have to reduce the costs of traceability (56% > 45%, N = 53). With most of them answering it is nice to have, with 33.96%, and with a numeric mean of 2.42 < 2.5[3].

2 A should have or must have for more guidance in the company regarding traceability (46% < 54%, N = 52). With most of them seeing it as a nice to have with 28.85% and with a numeric mean of 2.63 > 2.5.

3 A should have or must have for a more clear overview of the costs and benefits regarding traceability (disagree 32% < 68%, N= 53). With most of them should have with 45.28% and a numeric mean of 2.87 > 2.5.

4 A should have or must have to have easier access to information sources to be able to establish and maintain traceability (disagree 42% < 58%, N = 53). With most of them should have with 35.85% and a numeric mean of 2.64 < 2.5.

[3] Likert scale responses range from not needed, nice to have, should have, to must have. By changing these responses to their numeric variants, they correspond to 1-2-3-4 respectively. Mean of 2.5 is neutral, with means above indicating they perceive a certain need of more value or importance and means below indicating they perceive it of little value or little importance and more as optional.

5 A should have or must have to perform traceability in a more managed fashion (disagree 35% < 65%, N = 54). With most of them must have with 35.19% and a numeric mean of 2.89 > 2.5

6 A should have or must have to increase the awareness of the importance of traceability (disagree 36% < 64% agree, N = 55). With most of them must have with 35.55% and a numeric mean of 2.85 > 2.5.

7 Either not needed or nice to have for more staff, time, and resources to properly establish and maintain traceability (57% > 43%, N = 53). With most of them nice to have and should have with both 30.19% and a numeric mean of 2.30 > 2.5

8 A should have or must have for more collaboration between involved stakeholder teams regarding traceability (disagree 42% < agree 58%, N = 52). With most of them should have with 38.46% and a numeric mean of 2.65 > 2.5.

9 A should have or must have for more communication about who are responsible for traceability (disagree 37% < agree 63%, N = 52). With most of them should have with 38.46% and a numeric mean of 2.73.

10 Either not needed or nice to have for better traceability tools (disagree 51% > agree 49%, N = 53). With most of them nice to have with 30.19% and a numeric mean of 2.53 < 2.5 indicating that the participants more strongly disagree then strongly.

11 Either not needed or nice to have a more centralized development artifact repository to establish traceability more easily (disagree 60% > agree 40%, N = 55). With most of them nice to have with 32.73% and a numeric mean of 2.31 > 2.5

12 A should have or a must have for less complex traceability tools and easier integration with their existing tools (disagree 47% < agree 53%, N = 53). With most of them both nice to have and should have with 33.96% and a numeric mean of 2.58

13 A should have or a must have for more traceability automation (disagree 42% < agree 58%, N = 55). With most of them must have with 32.73% and a numeric mean of 2.82

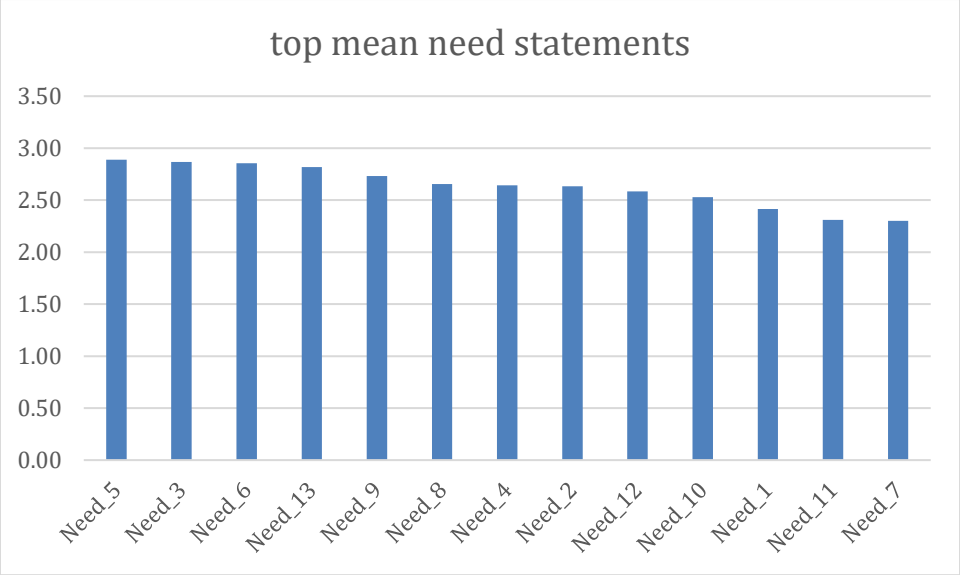


Figure 30: Perceived value of software traceability needs ranked from high value/importance to low value/importance based on mean.

By ranking the statements from high to low based on the means, in which the higher values indicate a higher perceived value and lower values indicate a lower perceived value, it is possible to see which needs the participants value the highest and value the least, as seen in figure 30.

In regards to needs that are of high value and importance, the results show that based on the means the participants highly value or have to most need for: Need_5: to *perform traceability in a more managed fashion*, Need_3: *for a more clear overview of the costs and benefits regarding traceability*, and Need_6: *to increase the awareness of the importance of traceability*.

In regards to needs that are of low priority and perceived value or importance, the results show that based on the means the participants value the following needs the least Need_7: *to have for more staff, time, and resources to properly establish and maintain traceability*, Need_11: *a more centralized development artifact repository to establish traceability more easily*, and Need_1: *to reduce the costs of traceability*.

6. Agile vs non-Agile software development

The following subsections describe the results for comparing Agile and non-Agile software development, in which the participants are grouped under Agile, Mixed, or Traditional software development based on their responses. Section 6.1 Describes descriptive results and compares the overall sentiment of the groups to each other via divergent bar charts regarding the current situation of software traceability and their perceived value of traceability in terms of similar as in section 5.2.

Section 6.2 describes the Likert response distribution, the characteristics, and tests these distributions with the use of Unpaired Two-Samples Wilcoxon tests, also referred to Mann-Whitney test, to check for significant differences in the Likert response distributions.

6.1 Questionnaire: descriptive results

6.1.1 Current situation

6.1.1.1 Management practices

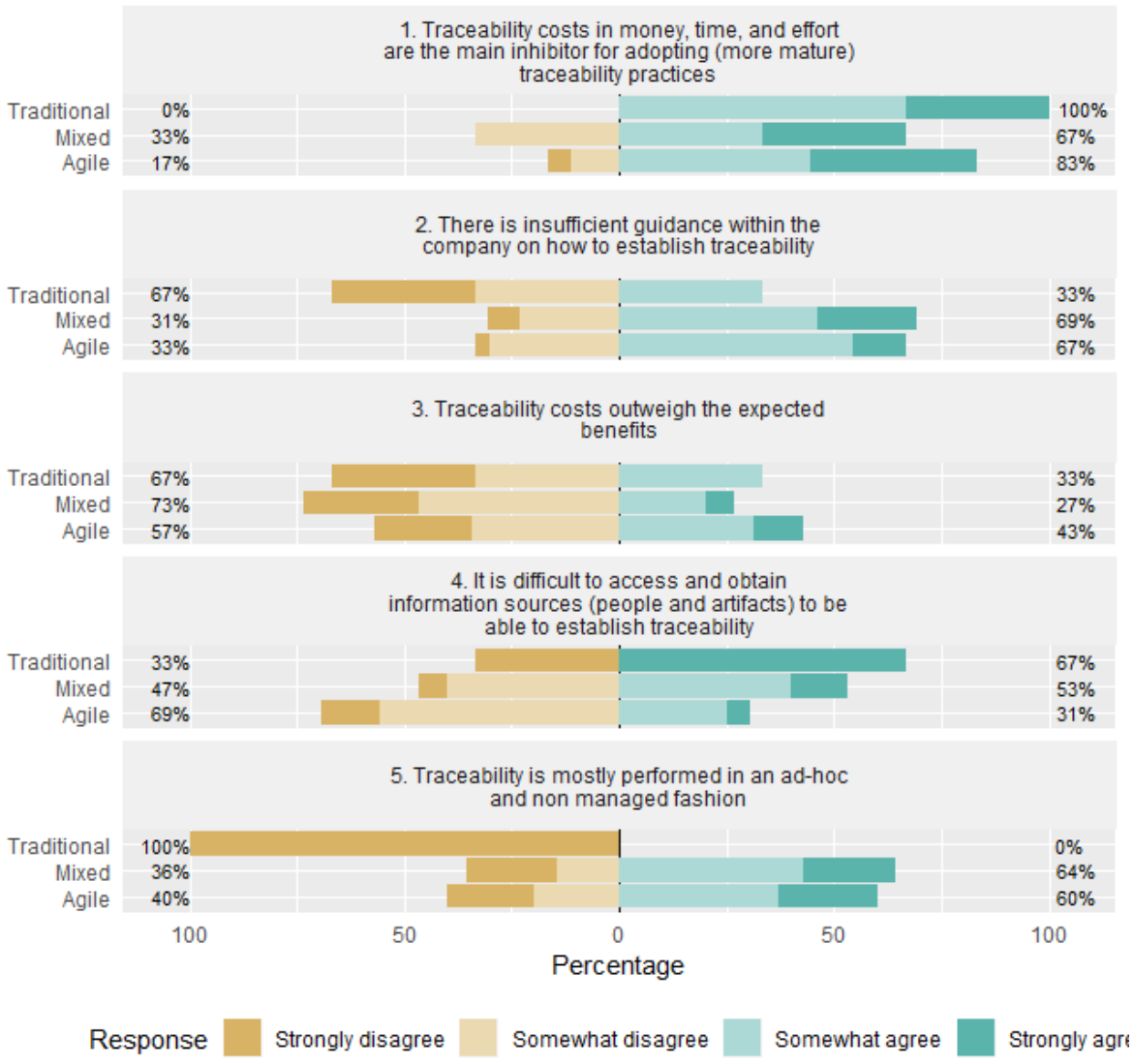


Figure 31: Divergent bar chart of overall sentiment regarding management statements of current situation between Agile, Mixed, and Traditional.

Based on the divergent bar chart as seen in figure 31, and the summary tables in appendix E.1, we see for each statement that:

1. Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices

    a. 100% of our participants that follow a Traditional development paradigm agree with the statement (disagree 0% < agree 100%, N = 3). With most of them somewhat agreeing, with 66.67%, followed by Strongly agree with 33.33%.

    b. 67% of our participants that follow a Mixed development approach agree with the statement (disagree 33% < 67%, N = 15). With somewhat disagree, somewhat agree and strongly agree having 33.33%.

    c. 83% of our participants that follow an Agile development approach agree with the statement (disagree 17% < 83%, N = 36). With most of them somewhat agreeing with 44.44%, followed by strongly agree with 38.89%, somewhat disagree with 11.11%, and strongly disagree with 5.56%

        Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of the three groups agree with the first statement.

2. There is insufficient guidance within the company on how to establish traceability

    a. 67% of our participants that follow a Traditional development paradigm disagree with the statement (disagree 67% < agree 33%, N = 3). With strongly disagreeing, somewhat disagreeing, and somewhat agree with 33.33%.

    b. 69% of our participants that follow a Mixed development approach agree with the statement (disagree 31% < 69%, N = 15). With most of them somewhat agreeing with 46.15%, followed by both somewhat disagree and strongly agree with 23.08%, and strongly disagree with 7.69%.

    c. 67% of our participants that follow an Agile development approach agree with the statement (disagree 33% < 67%, N = 33). With most of them somewhat agreeing with 54.55%, followed by somewhat disagree with 30.30%, strongly agree with 12.12%, and strongly disagree with 3.03%

        Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups in which most of the participants that follow a Traditional development process disagree with the second statement, while most of the participants of the other two groups agree with the second statement.

3. Traceability costs outweigh the expected benefits

   a. 67% of our participants that follow a Traditional development paradigm disagree with the statement (disagree 67% < agree 33%, N = 3). With strongly disagree, somewhat disagree, and somewhat agree, with 33.33%.

   b. 73% of our participants that follow a Mixed development approach disagree with the statement (disagree 73% < 23%, N = 15). With most of them somewhat disagreeing with 46.67%, followed by strongly disagree with 26.67%, somewhat agree with 20.00%, and strongly agree with 6.67%.

   c. 57% of our participants that follow an Agile development approach disagree with the statement (disagree 57% < 43%, N = 35). With most of them somewhat disagreeing with 34.29%, followed by somewhat agree with 31.43%, strongly disagree with 22.86%, and strongly agree with 11.43%.

      Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of the three groups disagree with the statement.

4. It is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability

   a. 67% of our participants that follow a Traditional development paradigm agree with the statement (disagree 33% < agree 67%, N = 3). With most of them strongly agreeing, with 66.67% followed by strongly disagree with 33.33%

   b. 53% of our participants that follow a Mixed development approach agree with the statement (disagree 47% < 53%, N = 15). With most of them somewhat disagreeing and somewhat agreeing with both 40%, followed by strongly agree with 13.33%, and strongly disagree with 6.67%

   c. 69% of our participants that follow an Agile development approach disagree with the statement (disagree 69% > 31%, N = 36). With most of them somewhat disagreeing with 55.56%, followed by somewhat agree with 25.00%, strongly disagree with 13.89%, and strongly agree with 5.56%

   Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups in which most of the participants that follow an Agile development process disagree with the fourth statement, while most of the participants of the other two groups agree with the statement.

5. Traceability is mostly performed in an ad-hoc and non managed fashion

   a. 100% of our participants that follow a Traditional development paradigm disagree with the statement (disagree 100% > agree 0%, N = 2). With all of them strongly disagreeing, with 100%.

b. 73% of our participants that follow a Mixed development approach agree with the statement (disagree 73% > 23%, N = 14). With most of them somewhat agreeing with 42.86%, followed by both strongly disagree and strongly agree with 21.43%, and somewhat disagree with 14.29%.

c. 57% of our participants that follow an Agile development approach agree with the statement (disagree 57% > 43%, N = 35). With most of them somewhat agreeing with 37.14%, followed by strongly agree with 22.86%, and both strongly disagree and somewhat agree with 20.00%

Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups in which most of the participants that follow a Traditional development process disagree with the fifth statement, while most of the participants of the other two groups agree with the statement.

6.1.1.2 Social collaboration practices

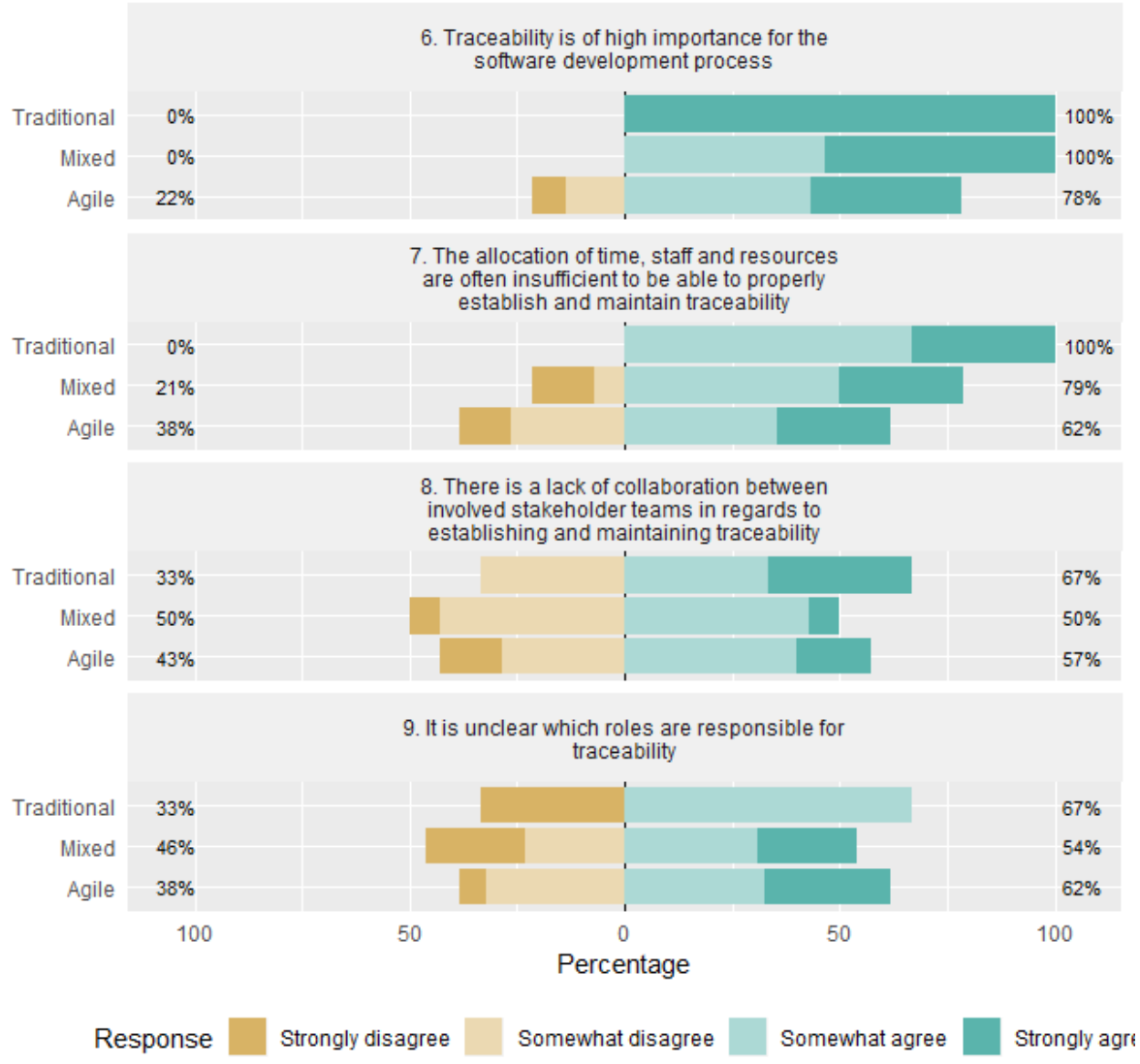


Figure 32: Divergent bar chart of overall sentiment regarding social statements of current situation between Agile, Mixed, and Traditional.

6. Traceability is of high importance for the software development process

    a. 100% of our participants that follow a Traditional development paradigm agree with the statement (disagree 0% < agree 100%, N = 3). With all of them strongly agreeing, with 100%.

    b. 100% of our participants that follow a Mixed development approach agree with the statement (disagree 0% < 100%, N = 15). With most of them strongly agreeing with 53.33%, followed by somewhat agree with 46.67%.

    c. 78% of our participants that follow an Agile development approach agree with the statement (disagree 22% < 78%, N = 37). With most of them somewhat agreeing with 43.24%, followed by strongly agree with 35.14%, somewhat disagree with 13.51%, and strongly disagree with 8.11%

    Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of the three groups agree with the statement.

7. The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability

    a. 100% of our participants that follow a Traditional development paradigm agree with the statement (disagree 0% < agree 100%, N = 3). With all of them somewhat agreeing with 66.67%, followed by strongly agree with 33.33%.

    b. 79% of our participants that follow a Mixed development approach agree with the statement (disagree 21% < 79%, N = 14). With most of them somewhat agreeing with 50.00%, followed by strongly agree with 28.57%, strongly disagree with 14.29%, and somewhat disagree with 7.14%.

    c. 62% of our participants that follow an Agile development approach agree with the statement (disagree 38% < 62%, N = 34). With most of them somewhat agreeing with 35.29%, followed by both somewhat disagree and strongly agree with 26.47%, and strongly disagree with 11.76%

    Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of the three groups agree with the statement.

8. There is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability

    a. 67% of our participants that follow a Traditional development paradigm agree with the statement (disagree 33% < agree 67%, N = 3). With somewhat disagree, somewhat agree, and strongly agree with 66.67%.

    b. 50% of our participants that follow a Mixed development approach disagree and agree with the statement (disagree 50% < 50%, N = 14). With most of them split between

somewhat disagreeing and somewhat agreeing with 42.86%, followed by both strongly disagree and strongly agree with 7.14%.

c. 57% of our participants that follow an Agile development approach agree with the statement (disagree 43% < 57%, N = 35). With most of them somewhat agreeing with 40.00%, followed by somewhat disagree with 28.57%, strongly agree with 17.14%, and strongly disagree with 14.29%.

Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups in which the participants that follow a Mixed development process were split evenly about the fifth statement, while most of the participants of the other two groups agree with the statement.

9. It is unclear which roles are responsible for traceability

a. 67% of our participants that follow a Traditional development paradigm agree with the statement (disagree 33% < agree 67%, N = 3). With most of them somewhat agreeing, with 66.67%, followed by strongly disagree with 33.33%.

b. 54% of our participants that follow a Mixed development approach agree with the statement (disagree 46% < 54%, N = 13). With most of them somewhat agreeing with 30.77%, followed by strongly disagree, somewhat disagree, and strongly agree with 23.08%.

c. 62% of our participants that follow an Agile development approach agree with the statement (disagree 38% < 62%, N = 34). With most of them somewhat agreeing and somewhat disagreeing with both 32.35%, followed by strongly agree with 29.41%, and strongly disagree with 5.88%

Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of the three groups agree with the statement.

6.1.1.3 Technical

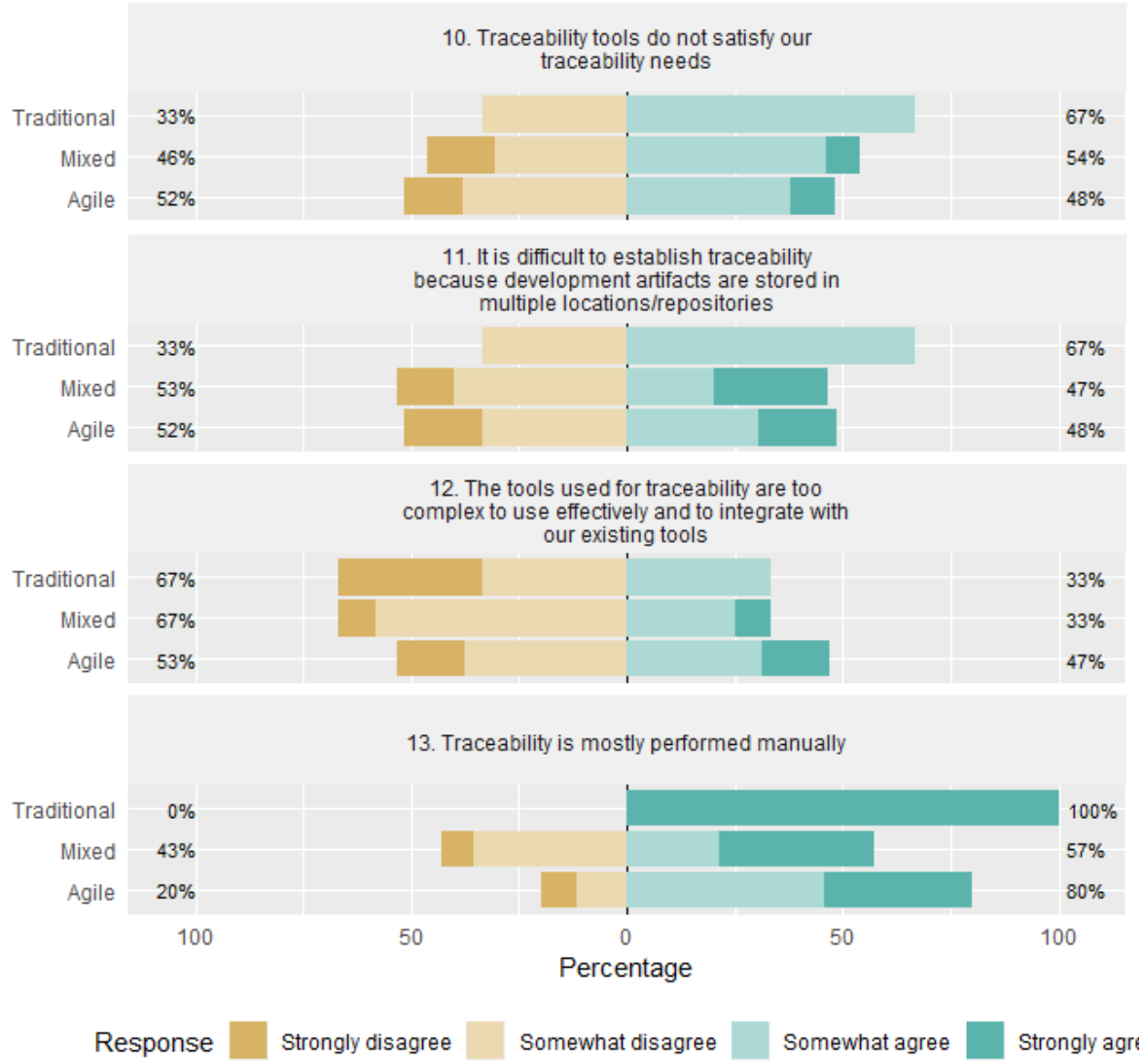


Figure 33: Divergent bar chart of overall sentiment regarding technical statements of current situation between Agile, Mixed, and Traditional.

10. Traceability tools do not satisfy our needs

a. 67% of our participants that follow a Traditional development paradigm agree with the statement (disagree 33% < agree 67%, N = 3). With most of them somewhat agreeing, with 66.67%, followed by somewhat disagree with 33.33%.

b. 54% of our participants that follow a Mixed development approach agree with the statement (disagree 46% < 54%, N = 13). With most of them somewhat agreeing with 46.15%, followed by somewhat disagree with 30.77%, strongly disagree with 15.38%, and strongly agree with 7.69%.

c. 52% of our participants that follow an Agile development approach disagree with the statement (disagree 52% < 48%, N = 29). With both somewhat disagreeing and somewhat agree with 37.93%, followed by strongly disagree with 13.79%, and strongly agree with 10.34%.

Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups in which the participants that follow an Agile development process disagree with the fifth statement, while most of the participants of the other two groups agree with the statement.

11. It is difficult to establish traceability because development artifacts are stored in multiple locations/repositories

    a. 67% of our participants that follow a Traditional development paradigm agree with the statement (disagree 33% < agree 67%, N = 3). With most of them somewhat agreeing, with 66.67%, followed by somewhat disagree with 33.33%.

    b. 53% of our participants that follow a Mixed development approach disagree with the statement (disagree 53% < 47%, N = 15). With most of them somewhat disagreeing with 40%, followed by strongly agree with 26.67%, somewhat agree with 20.00%, and strongly disagree with 13.33%.

    c. 52% of our participants that follow an Agile development approach disagree with the statement (disagree 52% < 48%, N = 3). With most of them somewhat disagreeing with 33.33%, followed by somewhat agree with 30.30%, and both strongly disagree and strongly agree with 18.18%.

    Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups in which the participants that follow a Traditional development process agree with the statement, while most of the participants of the other two groups disagree with the statement.

12. Tools used for traceability are too complex to effectively use and to integrate with our existing tools

    a. 67% of our participants that follow a Traditional development paradigm disagree with the statement (disagree 67% < agree 33%, N = 3). With strongly disagree, somewhat disagree, and somewhat agree with 33.33%.

    b. 67% of our participants that follow a Mixed development approach disagree with the statement (disagree 67% < 33%, N = 12). With most of them somewhat disagreeing with 58.33%, followed by somewhat agree with 25.00%, and both strongly disagree and strongly agree with 8.33%.

    c. 53% of our participants that follow an Agile development approach disagree with the statement (disagree 53% < 47%, N = 32). With most of them somewhat disagreeing with 37.50%, followed by somewhat agree with 31.25%, and both strongly disagree and strongly agree with 15.63%.

    Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of the three groups agree with the statement.

13. Traceability is mostly performed manually

a. 100% of our participants that follow a Traditional development paradigm agree with the statement (disagree 0% < agree 100%, N = 3). With most of them strongly agreeing, with 100%.

b. 57% of our participants that follow a Mixed development approach agree with the statement (disagree 43% < 57%, N = 14). With most of them somewhat disagreeing and strongly agreeing both with 35.70%, followed by somewhat agree with 21.43%, and strongly disagree with 7.14%.

c. 80% of our participants that follow an Agile development approach agree with the statement (disagree 20% < 80%, N = 35). With most of them somewhat agreeing with 45.71%, followed by strongly agree with 34.29%, somewhat disagree with 11.43%, and strongly disagree with 8.57%.

Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of the three groups agree with the statement.

Current				
Statement		Agile	Mixed	Traditional
Current_1	Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices			
Current_2	There is insufficient guidance within the company on how to establish traceability			
Current_3	Traceability costs outweigh the expected benefits			
Current_4	It is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability			
Current_5	Traceability is mostly performed in an ad-hoc and non managed fashion			
Current_6	Traceability is of high importance for the software development process			
Current_7	The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability			
Current_8	There is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability			
Current_9	It is unclear which roles are responsible for traceability			
Current_10	Traceability tools do not satisfy our traceability needs			
Current_11	It is difficult to establish traceability because development artifacts are stored in multiple locations/repositories			
Current_12	The tools used for traceability are too complex to use effectively and to integrate with our existing tools			
Current_13	Traceability is mostly performed manually			

Table 20: Summary sentiment table

Based on the divergent bar chart results and the summarization in table 20, the results show no difference between the sentiment of the groups for the following current situation statements:

- Current_1: Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices
- Current_3: Traceability costs outweigh the expected benefits
- Current_6: Traceability is of high importance for the software development process
- Current_7: The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability
- Current_9: It is unclear which roles are responsible for traceability
- Current 12, The tools used for traceability are too complex to use effectively and to integrate with our existing tools
- Current_13, Traceability is mostly performed manually

The results do show differences regarding the sentiment between the groups for:

- Current_2: There is insufficient guidance within the company on how to establish traceability
- Current_4: It is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability
- Current_5: Traceability is mostly performed in an ad-hoc and non managed fashion
- Current_8: There is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability
- Current_10: Traceability tools do not satisfy our traceability needs
- Current_11: It is difficult to establish traceability because development artifacts are stored in multiple locations/repositories

6.1.1.4 Comparing numeric means

Furthermore, when ranking the perceived current situation statements based on their means, as shown in figures 34-36, the results show the top three current situation statements the groups agree on:

- Agile
  - Current_1: Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices
  - Current_13: Traceability is mostly performed manually
  - Current_6: Traceability is of high importance for the software development process
- Mixed
  - Current_6: Traceability is of high importance for the software development process
  - Current_1: Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices
  - Current_7: The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability
- Traditional
  - Current_6: Traceability is of high importance for the software development process
  - Current_13: Traceability is mostly performed manually
  - Current_1: Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices

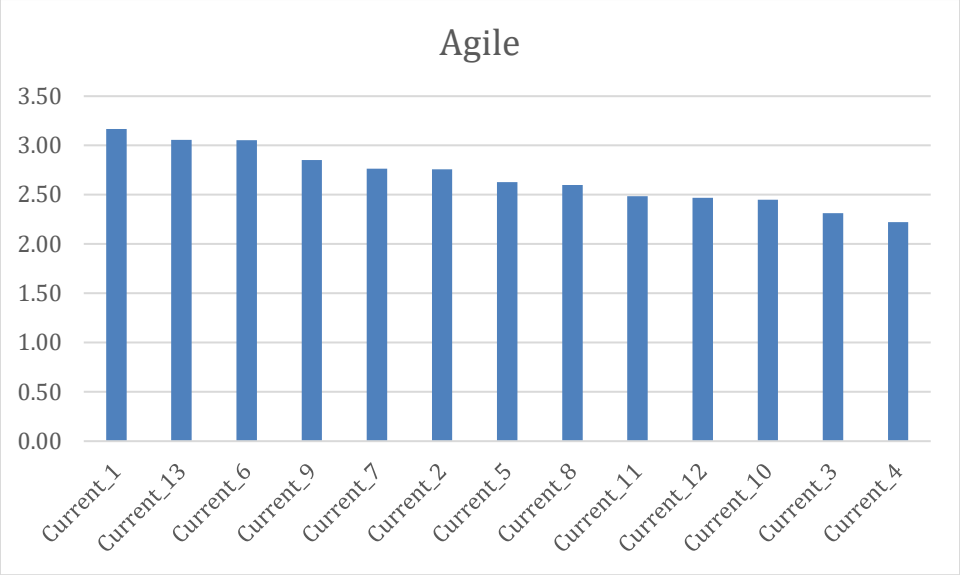


Figure 34: Ranked means of current statements of Agile development

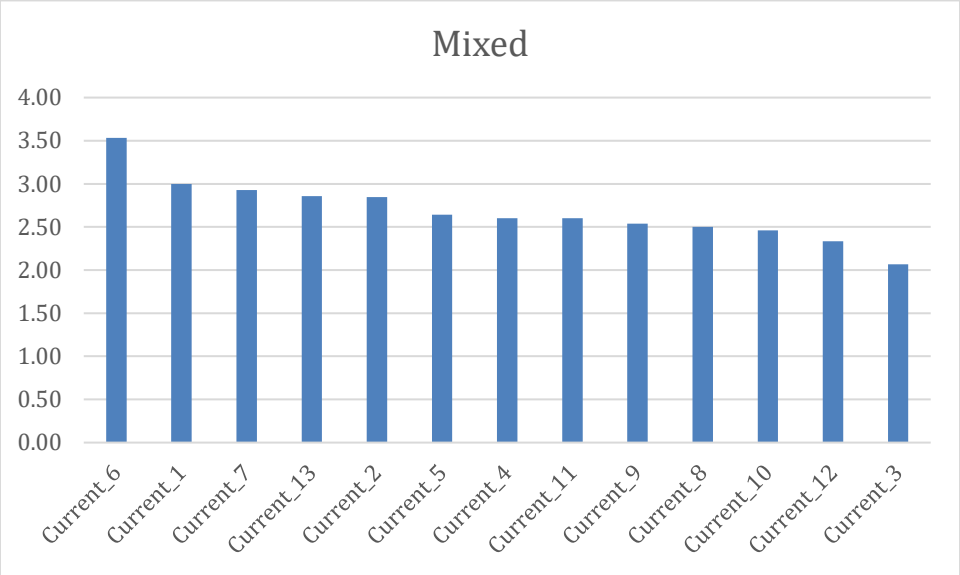


Figure 35: Ranked means of current statements for mixed development

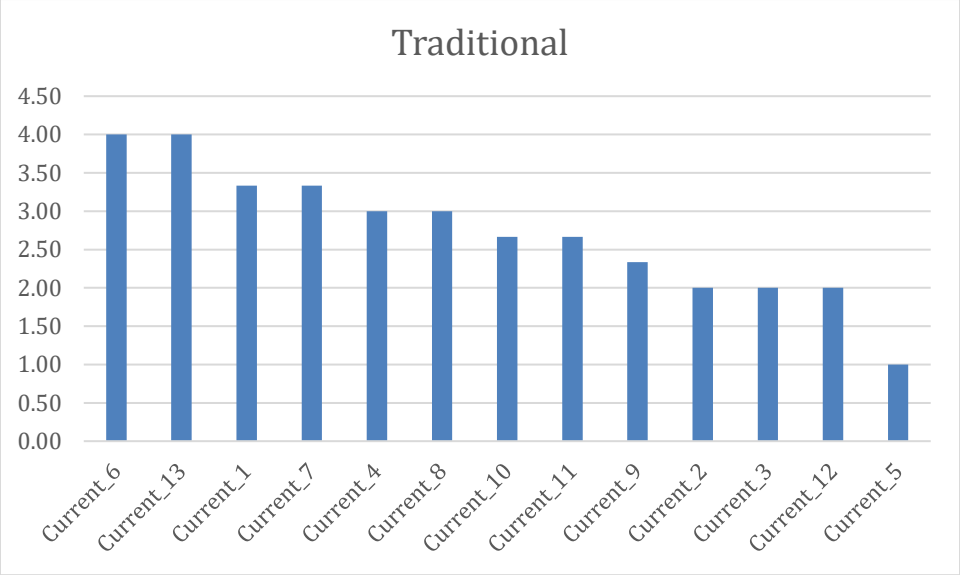


Figure 36: Ranked means of current statements for traditional development

6.2.2 Needs

6.2.2.1 Management

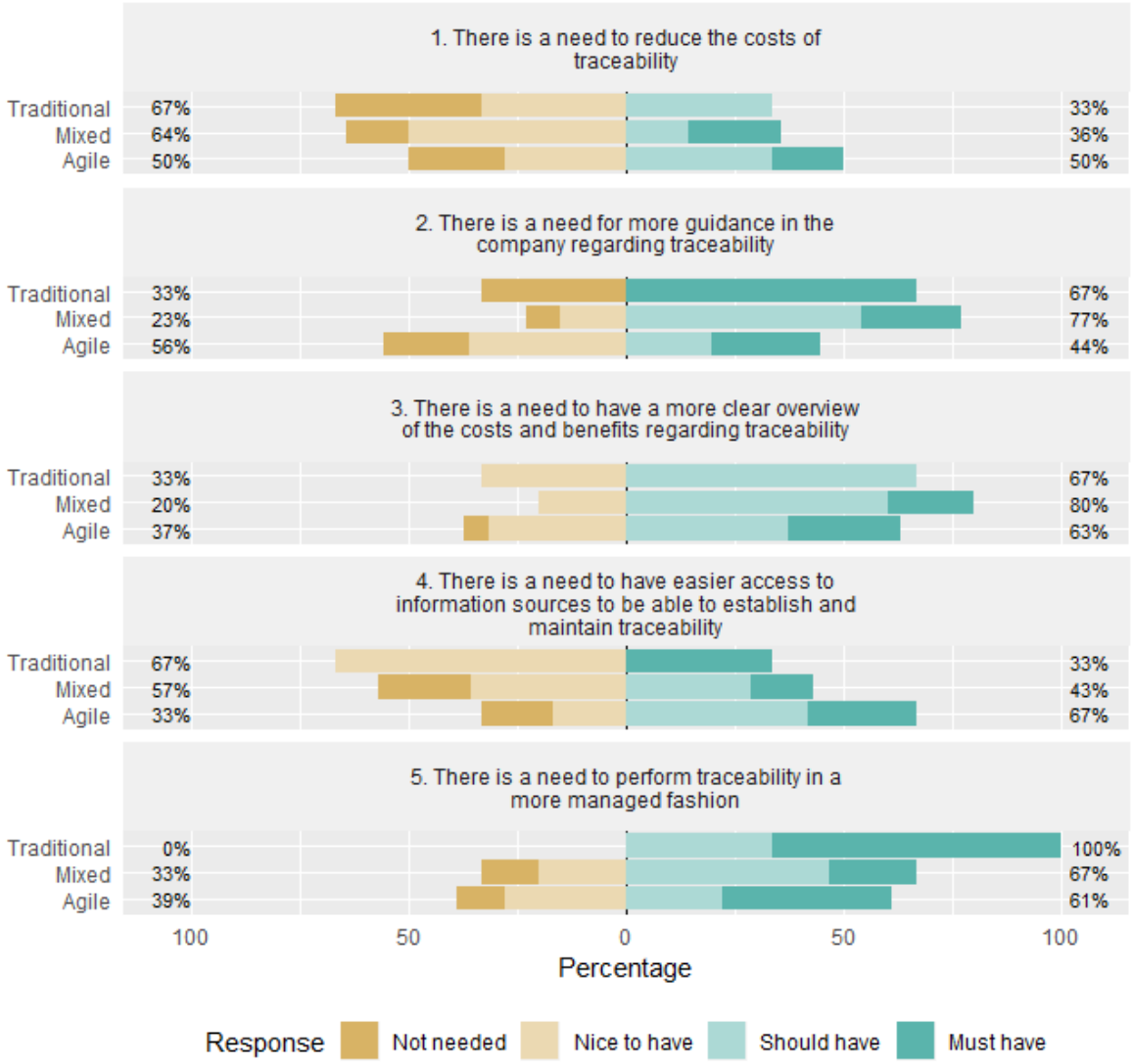


Figure 37: *Divergent bar chart of overall sentiment regarding management need statements between Agile, Mixed, and Traditional.*

1. There is a need to reduce the costs of traceability

a. 67% of our participants that follow a Traditional development paradigm see the need as not important (not important 67% > important 33%, N = 3). With not needed, nice to have, and must have equally at 33%, followed by must have with 0%.

b. 64% of our participants that follow a Mixed development approach see the need as not important (not important 64% > important 36%, N = 14). With most of them seeing it as nice to have with 50.00%, followed by must have 21.4%, and both not needed and should have with 14.29%.

c. 50% of our participants that follow an Agile development approach see the need as not important while the other 50% see it as important (50% = 50%, N = 32). With most of them

seeing it as a should have with 33.33%, followed by nice to have with 27.78%, not needed with 22.22%, must have with 16.67.

Based on these results, there is a difference in the overall sentiment regarding the need between the groups in which the participants that follow an Agile software development process are equally split over the importance of the need, while most of the participants of the other two groups see the need as not important.

2. There is a need for more guidance in the company regarding traceability

a. 67% of our participants that follow a Traditional development paradigm see the need as important (not important 33% < important 67%, N = 3). With must have at 66.67%, followed by not needed with 33,33%.

b. 77% of our participants that follow a Mixed development approach see the need as important (not important 23% < important 77%, N = 13). With most of them seeing it as should have with 53.85%, followed by must have 23.08%, Nice to have with 15.38%, not needed with 7.69%.

c. 56% of our participants that follow an Agile development approach see the need as not important (56% > 44%, N = 36). With most of them seeing it as nice to have with 36.11%, followed by must have with 25.00%, and both not needed and should have with 19.44.

Based on these results, there is a difference in the overall sentiment regarding the need between the groups in which the participants that follow an Agile software development process see the need as not important, while most of the participants of the other two groups see the need as important.

3. There is a need for a more clear overview of the costs and benefits regarding traceability

a. 67% of our participants that follow a Traditional development paradigm see the need as important (not important 33% < important 67%, N = 3). With should have at 66.67%, followed by nice to have with 33,33%.

b. 80% of our participants that follow a Mixed development approach see the need as important (not important 20% < important 80%, N = 15). With most of them seeing it as should have with 60.00%, followed by both nice to have and must have with 20.00%.

c. 63% of our participants that follow an Agile development approach see the need as important (37% > 63%, N = 36). With most of them seeing it as should have with 37.14%, followed by nice to have with 31.43%, must have with 25.71, not needed with 5.71%.

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of the three groups see the need as important.

4. There is a need to have easier access to information sources to be able to establish and maintain traceability

a. 67% of our participants that follow a Traditional development paradigm see the need as not important (not important 67% < important 33%, N = 3). With nice to have at 66.67%, followed by must to have with 33,33%.

b. 57% of our participants that follow a Mixed development approach see the need as not important (not important 57% < important 43%, N = 14). With most of them seeing it as nice to have with 35.71%, followed by should have with 28.57%, not needed with 21.43%, must have with 14.29%.

c. 67% of our participants that follow an Agile development approach see the need as important (33% > 67%, N = 36). With most of them seeing it as should have with 41.67%, followed by must have with 25.00%, and both not needed and nice to have with both 16.67%.

Based on these results, there is a difference in the overall sentiment regarding the need between the groups in which the participants that follow an Agile software development process see the need as important, while most of the participants of the other two groups see the need as not important.

5. There is a need to perform traceability in a more managed fashion

a. 100% of our participants that follow a Traditional development paradigm see the need as important (not important 0% < important 100%, N = 3). With must have at 66.67%, followed by should have with 33,33%.

b. 67% of our participants that follow a Mixed development approach see the need as important (not important 33% < important 67%, N = 15). With most of them seeing it as a should have with 46.67%, followed by both nice to have and must have with 20.00%, not needed with 13.33%.

c. 61% of our participants that follow an Agile development approach see the need as important (39% > 61%, N = 36). With most of them seeing it as a must have with 38.89%, followed by nice to have with 27.78%, should have with 22.22%, not needed with 11.11%.

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of the three groups see the need as important.

6.2.2.2 Social

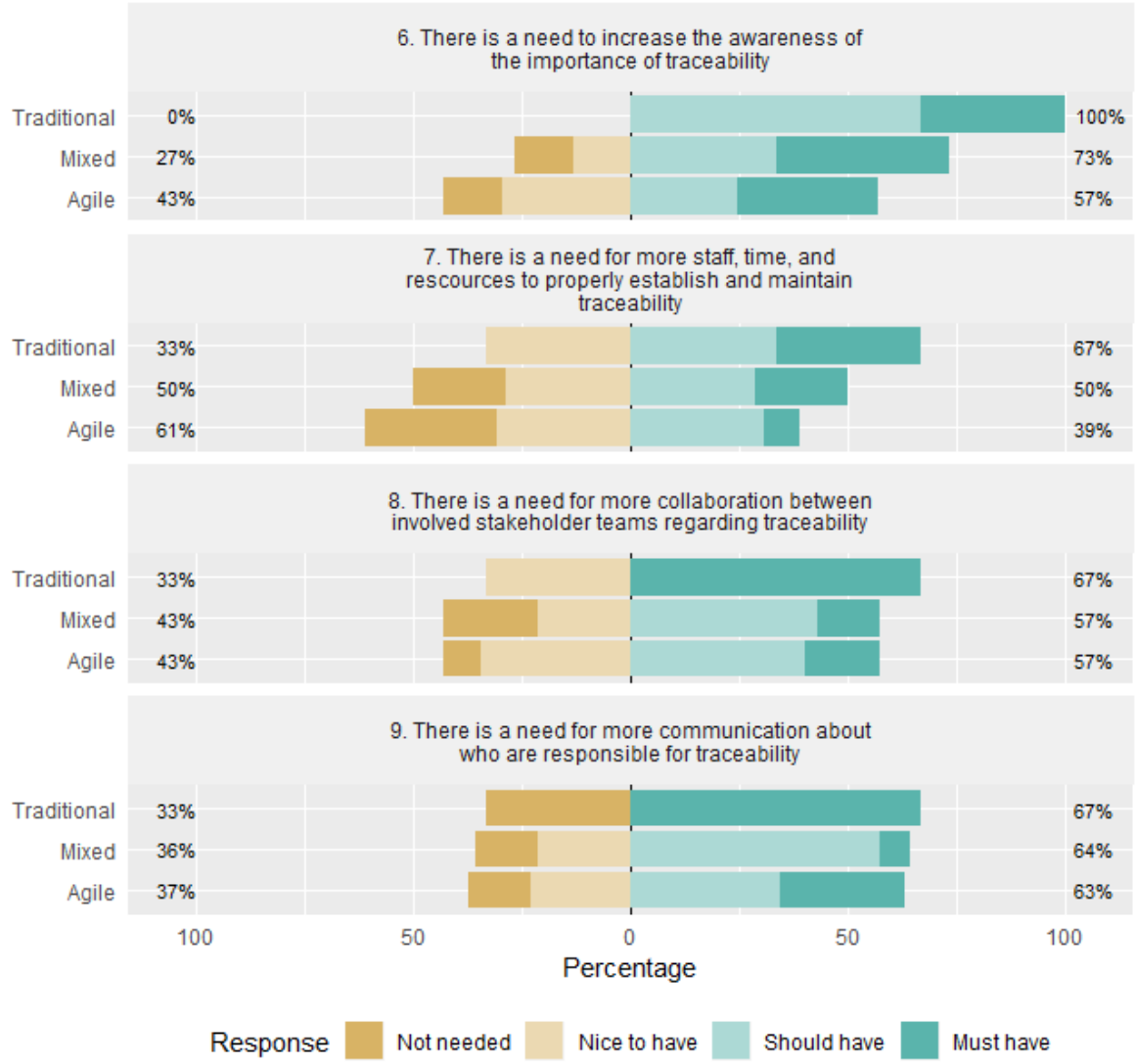


Figure 38: *Divergent bar chart of overall sentiment regarding social need statements between Agile, Mixed, and Traditional.*

6. There is a need to increase the awareness of the importance of traceability

a. 100% of our participants that follow a Traditional development paradigm see the need as important (not important 0% < important 100%, N = 3). With should have at 66.67%, followed by must have with 33,33%.

b. 73% of our participants that follow a Mixed development approach see the need as important (not important 27% < important 73%, N = 15). With most of them seeing it as a must have with 40.00%, followed by should have with 33.33%, and both not needed and nice to have with 13.33%.

c. 57% of our participants that follow an Agile development approach see the need as important (43% > 57%, N = 36). With most of them seeing it as a must have with 32.43%, followed by nice to have with 29.73%, should have with 24.32%, not needed with 13.51%.

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of the three groups see the need as important.

7. There is a need for more staff, time, and resources to properly establish and maintain traceability

a. 67% of our participants that follow a Traditional development paradigm see the need as important (not important 33% < important 67%, N = 3). With nice to have, should have, and must have equally at 33.33%.

b. 50% of our participants that follow a Mixed development approach see the need as important and the other 50% as not important (not important 50% < important 50%, N = 14). With both should have and must have with 28.57%, followed by both not needed and must have at 21.43%, and both not needed and nice to have with 13.33%.

c. 61% of our participants that follow an Agile development approach see the need as not important (61% > 39%, N = 36). With not needed, nice to have, and should have equally at 30.56%, followed by must have at 8.33%.

Based on these results, there is a difference in the overall sentiment regarding the need between the groups in which the participants that follow a Traditional software development process see the need as important, while the participants that follow a Mixed software development process are equally split, and while the participants that follow an Agile software development process see the need as not important.

8. There is a need for more collaboration between involved stakeholder teams regarding traceability

a. 67% of our participants that follow a Traditional development paradigm see the need as important (not important 33% < important 67% N = 3). With must have at 66.67%, followed by nice to have with 33,33%.

b. 57% of our participants that follow a Mixed development approach see the need as important (not important 43% < important 57%, N = 14). With most of them seeing it as a should have with 42.86%, followed by both not needed and nice to have with 21.43%, must have with 14.29%.

c. 57% of our participants that follow an Agile development approach see the need as important (43% > 57%, N = 35). With most of them seeing it as a should have with 40.00%, followed by nice to have with 34.29%, must have with 17.14%, not needed with 8.57%.

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of the three groups see the need as important.

9. There is a need for more communication about who are responsible for traceability

a. 67% of our participants that follow a Traditional development paradigm see the need as important (not important 33% < important 67% N = 3). With must have at 66.67%, followed by not needed with 33,33%.

b. 64% of our participants that follow a Mixed development approach see the need as important (not important 36% < important 64%, N = 14). With most of them seeing it as a should have with 57.14%, followed by nice to have with 21.43%, nice to have with 14.29%. not needed with 7.14%.

c. 63% of our participants that follow an Agile development approach see the need as important (37% > 63%, N = 35). With most of them seeing it as a should have with 34.29%, followed by must have with 28.57%, nice to have with 22.86%, not needed with 14.29%.

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of the three groups see the need as important.

6.2.2.3 Technical

Figure 39: *Divergent bar chart of overall sentiment regarding technical need statements between Agile, Mixed, and Traditional.*

10. There is a need for better traceability tools

a. 67% of our participants that follow a Traditional development paradigm see the need as important (not important 33% < important 67% N = 3). With should have at 66.67%, followed by nice to have with 33,33%.

b. 67% of our participants that follow a Mixed development approach see the need as not important (not important 67% < important 33%, N = 15). With most of them seeing it as a nice to have with 46.67%, followed by must have with 26.67%, not needed with 20.00%, nice to have with 6.67%.

c. 54% of our participants that follow an Agile development approach see the need as important (46% > 54%, N = 35). With most of them seeing it as a should have with 28.57%, followed by must have with 25.71%, and both not needed and nice to have with 22.86%.

Based on these results, there is a difference in the overall sentiment regarding the need between the groups in which the participants that follow a Mixed software development process see the need as not important, while most of the participants of the other two groups see the need as important.

11. There is a need to have a more centralized development artifact repository to establish traceability more easily

a. 67% of our participants that follow a Traditional development paradigm see the need as important (not important 33% < important 67% N = 3). With nice to have, should have, and must have at 33.33%.

b. 53% of our participants that follow a Mixed development approach see the need as not important (not important 53% < important 47%, N = 15). With not needed, nice to have, and should have at 26.67%, followed by must have at 20.00%.

c. 65% of our participants that follow an Agile development approach see the need as not important (65% > 35%, N = 37). With most of them seeing it as a nice to have with 35.14%, followed by not needed with 29.73%, should have with 18.92%, must have at 16.22%.

Based on these results, there is a difference in the overall sentiment regarding the need between the groups in which the participants that follow a Traditional software development process see the need as important, while most of the participants of the other two groups see the need as not important.

12. There is a need for less complex traceability tools and easier integration with their existing tools

a. 100% of our participants that follow a Traditional development paradigm see the need as important (not important 0% < important 100% N = 3). With should have at 66.67%, followed by must have at 33.33%.

b. 60% of our participants that follow a Mixed development approach see the need as not important (not important 60% < important 40%, N = 15). With most of them seeing it as a nice to have with 40%, followed by should have with 33.33%, not needed with 20%, must have with 6.67%.

c. 54% of our participants that follow an Agile development approach see the need as important (46% > 54%, N = 35). With most of them seeing it as a nice to have with 34.29%, followed by should have with 31.43%, must have with 22.86%, not needed at 11.43%.

Based on these results, there is a difference in the overall sentiment regarding the need between the groups in which the participants that follow a Mixed software development

process see the need as not important, while most of the participants of the other two groups see the need as important.

13. There is a need for more traceability automation

a. 67% of our participants that follow a Traditional development paradigm see the need as important (not important 33% < important 67% N = 3). With nice to have, should have, must have at 33.33%.

b. 60% of our participants that follow a Mixed development approach see the need as important (not important 40% < important 60%, N = 15). With most of them seeing it as a must have with 40%, followed by nice to have with 26.67%, should have with 20.00%, not needed with 13.33%.

c. 57% of our participants that follow an Agile development approach see the need as important (43% > 57%, N = 37). With most of them seeing it as a nice to have with 35.14%, followed by must have with 29.73%, should have with 27.03%, not needed at 0%.

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of the three groups see the need as important.

Need				
Statement		Agile	Mixed	Traditional
Need_1	There is a need to reduce the costs of traceability			
Need_2	There is a need for more guidance in the company regarding traceability			
Need_3	There is a need to have a more clear overview of the costs and benefits regarding traceability			
Need_4	There is a need to have easier access to information sources to be able to establish and maintain traceability			
Need_5	There is a need to perform traceability in a more managed fashion			
Need_6	There is a need to increase the awareness of the importance of traceability			
Need_7	There is a need for more staff, time, and resources to properly establish and maintain traceability			
Need_8	There is a need for more collaboration between involved stakeholder teams regarding traceability			
Need_9	There is a need for more communication about who are responsible for traceability			
Need_10	There is a need for better traceability tools			
Need_11	There is a need for a more centralized development artifact repository to establish traceability more easily			
Need_12	There is a need for less complex traceability tools and easier integration with our existing tools			
Need_13	There is a need for more traceability automation			

Table 21: Sentiment summary table: Agile vs Mixed vs Traditional: needs. Yellow denotes that most of the participants of a specific group do not see the need of high importance. Blue denotes that most of the participants see the need of high importance and priority. White denotes that the sentiment of the participants are split evenly.

Based on the divergent bar chart results and the summarization in table 21, the results show no difference between the sentiment of the groups for the following needs:

- Need_1: There is a need to reduce the costs of traceability.
- Need_3: There is a need for a more clear overview of the costs and benefits regarding traceability
- Need_5: There is a need to perform traceability in a more managed fashion
- Need_6: There is a need to increase the awareness of the importance of traceability
- Need_8, There is a need for more collaboration between involved stakeholder teams regarding traceability
- Need_9, There is a need for more communication about who are responsible for traceability
- Need_13. There is a need for more traceability automation

The results do show differences regarding the sentiment between the groups for:

- Need_2: There is a need for more guidance in the company regarding traceability
- Need_4: There is a need to have easier access to information sources to be able to establish and maintain traceability
- Need_7: There is a need for more staff, time, and resources to properly establish and maintain traceability
- Need_10: There is a need for better traceability tools
- Need_11: There is a need for a more centralized development artifact repository to establish traceability more easily
- Need_12: There is a need for less complex traceability tools and easier integration with our existing tools

6.2.2.4 Comparing numeric means

Furthermore, when looking at the ranked needs based on their means, as shown in figures 40-42, the results show the top three highest priority or most valued needs per group:

- Agile
  - Need_5: *There is a need to perform traceability in a more managed fashion*
  - Need_3: *There is a need for a more clear overview of the costs and benefits regarding traceability*
  - Need_13: *There is a need for more traceability automation*
- Mixed
  - Need_3: *There is a need for a more clear overview of the costs and benefits regarding traceability*
  - Need_6: *There is a need to increase the awareness of the importance of traceability*
  - Need_2: There is a need for more guidance in the company regarding traceability
- Traditional
  - Need_5: *There is a need to perform traceability in a more managed fashion*
  - Need_6: *There is a need to increase the awareness of the importance of traceability*
  - Need_8: *There is a need for more collaboration between involved stakeholder teams regarding traceability*

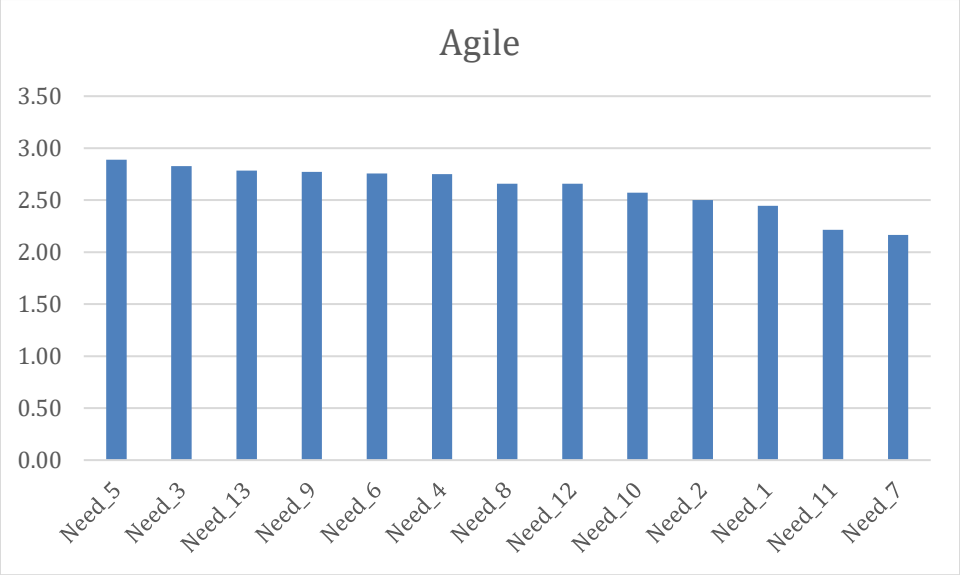


Figure 40: ranked means need Agile

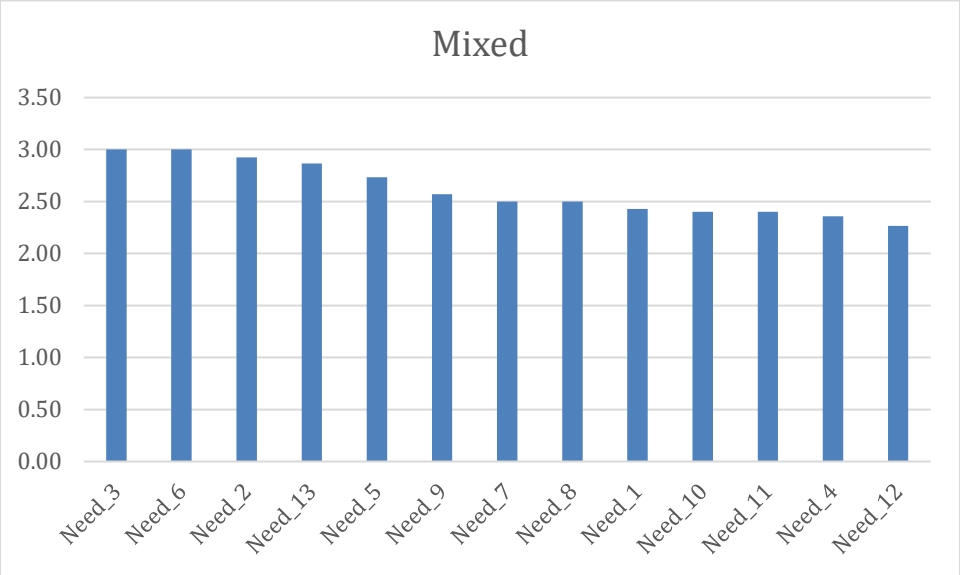


Figure 41: ranked means need Mixed

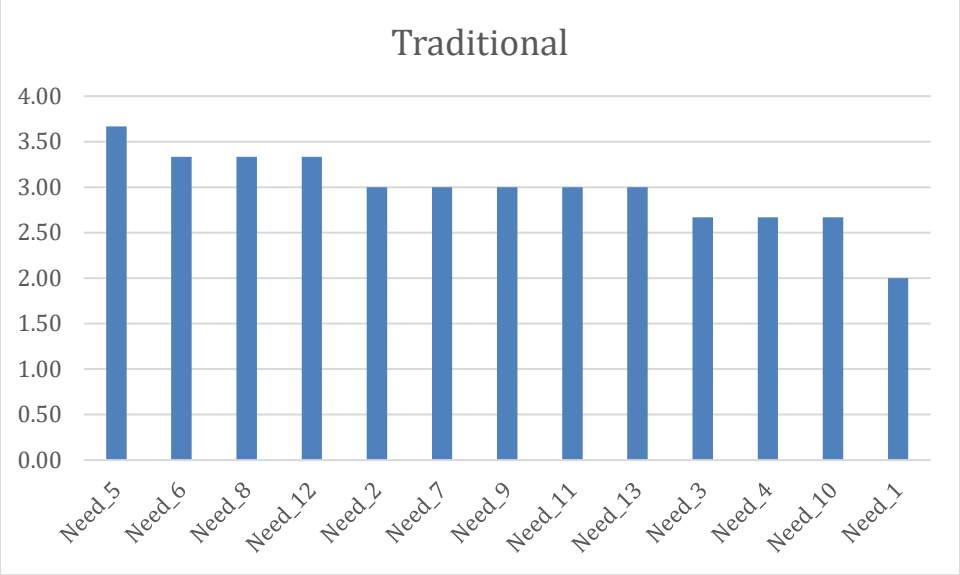


Figure 42: ranked means need Traditional

6.2 Questionnaire: statistical analysis

Recall that the responses on our survey are based on Likert scales, in which it is typical to use nonparametric tests for these types of data. In addition, as seen in appendix E.1.6, the distributions of the groups per statement do not follow a normal bell curve distribution for every statement, which is typically assumed for parametric tests.

Considering that our data does not follow a normal distribution, is nonparametric and unpaired, for statistically testing for any significant differences between Agile and non-Agile, we will be looking at the distributions by using an unpaired two-samples Wilcoxon test, (also referred to Mann-Whitney test). These tests are used to analyze and compare the distributions of the Likert responses and to see if there are any significant differences between the distributions of the groups. The results are summarized in the tables as seen in section 6.2.1 and 6.2.2.

For each of these tables, the left most column contains the statement, followed by three columns which show the effect size between the groups, in which the effect size is the difference between the means. For example, for statement current_1, column Agile_Mixed we can see that Agile agrees or disagrees more strongly than Mixed by a margin of 0.17. The blue and yellow fill colors denote a positive or negative difference by comparing the first group to the second group as such: group1_group2, in which a blue column indicates that group1 either more strongly agrees or disagrees compared to group2. The last three columns contain the p values of the Wilcoxon test, in which a = 0.05, and any p-value less than 0.05 indicates a significant difference in the Likert response distribution between the compared groups.

6.2.1 Unpaired Two-Samples Wilcoxon test: current situation

Recall, as described in section 4.1, the null and alternative hypotheses for the perceived current situation between Agile, non-Agile:

RQ4.1	*"Is the current software traceability situation affected by the type of software development life cycle?"*
H0	*The current software traceability situation is not affected by the type of software development life cycle*

H1	*The current software traceability situation is affected by the type of software development life cycle*

Based on the results as shown in table 22, we see only one case of a significant difference in the Likert response distribution (a=0.05) which is between Agile and Traditional for statement 13 regarding the current situation: *traceability is mostly performed manually*. The results show that the Traditional group agrees or disagrees more strongly compared to Agile with 0.94 difference between the means and a p-value of 0.048, ($p < 0.05$). Meaning the null hypothesis is rejected for statement 13 regarding the perceived current situation of software traceability.

Statements	Effect Size			p-values		
Statement	Agile_Mixed	Agile_Traditional	Mixed_Traditional	Agile_Mixed 2	Agile_Traditional 2	Mixed_Traditional 2
Current_1	0.17	-0.17	-0.33	0.445	0.886	0.571
Current_2	-0.09	0.76	0.85	0.659	0.156	0.201
Current_3	0.25	0.31	0.07	0.393	0.630	1.000
Current_4	-0.38	-0.78	-0.40	0.120	0.303	0.496
Current_5	-0.01	1.63	1.64	0.963	0.055	0.081
Current_6	-0.48	-0.95	-0.47	0.087	0.051	0.161
Current_7	-0.16	-0.57	-0.40	0.552	0.351	0.628
Current_8	0.10	-0.40	-0.50	0.673	0.532	0.415
Current_9	0.31	0.52	0.21	0.399	0.466	0.834
Current_10	-0.01	-0.22	-0.21	0.919	0.654	0.770
Current_11	-0.12	-0.18	-0.07	0.763	0.742	0.901
Current_12	0.14	0.47	0.33	0.645	0.459	0.635
Current_13	0.20	-0.94	-1.14	0.512	0.048	0.079

Table 22: Wilcox analysis results current Agile vs Mixed vs Traditional

Based on this we can conclude from a quantitative perspective that the data results in one significant difference between Agile and non-Agile, which is statement 13 regarding the perceived current situation, in which there is a significant difference in the Likert response distribution between Agile and Traditional development in regards to the statement that traceability is mostly performed manually.

6.2.2 Unpaired Two-Samples Wilcoxon test: need

Recall, as described in section 4.1, the null and alternative hypotheses for the perceived value of software traceability needs between Agile, non-Agile:

RQ4.2	*"Are the software traceability needs affected by the type of software development life cycle?"*
H0	*The software traceability needs are not affected by the type of software development life cycle*
H1	*The software traceability needs are affected by the type of software development life cycle*

Based on the results as shown in table 23, it is not possible to reject the null hypothesis. Indicating that there is no quantitative significant difference (a=0.05) regarding the perceived value and priority

of the needs between Agile, Mixed, and Traditional software development processes according to the Wilcoxon test.

Statements	Effect size			p-values		
Statement	Agile_Mixed	Agile_Traditional	Mixed_Traditional	Agile_Mixed	Agile_Traditional	Mixed_Traditional
Need_1	0.02	0.44	0.43	0.902	0.493	0.592
Need_2	-0.42	-0.50	-0.08	0.196	0.511	0.617
Need_3	-0.17	0.16	0.33	0.550	0.753	0.454
Need_4	0.39	0.08	-0.31	0.203	0.826	0.741
Need_5	0.16	-0.78	-0.93	0.583	0.233	0.116
Need_6	-0.24	-0.58	-0.33	0.442	0.392	0.801
Need_7	-0.33	-0.83	-0.50	0.328	0.187	0.514
Need_8	0.16	-0.68	-0.83	0.691	0.254	0.239
Need_9	0.20	-0.23	-0.43	0.478	0.593	0.383
Need_10	0.17	-0.10	-0.27	0.614	0.955	0.573
Need_11	-0.18	-0.78	-0.60	0.585	0.219	0.427
Need_12	0.39	-0.68	-1.07	0.203	0.235	0.070
Need_13	-0.08	-0.22	-0.13	0.751	0.747	0.950

Table 23: Wilcox analysis results needs Agile vs Mixed vs Traditional

Based on this we can conclude that there are no significant differences in regarding the Likert response distribution between Agile and non-Agile groups regarding the perceived value and priority of the needs of software traceability.

7. Safety vs non-Safety critical projects

This section describes the quantitative differences between safety critical and non-safety critical respondents

7.1 Questionnaire: descriptive results

7.1.1 Current situation

7.1.1.1 Management

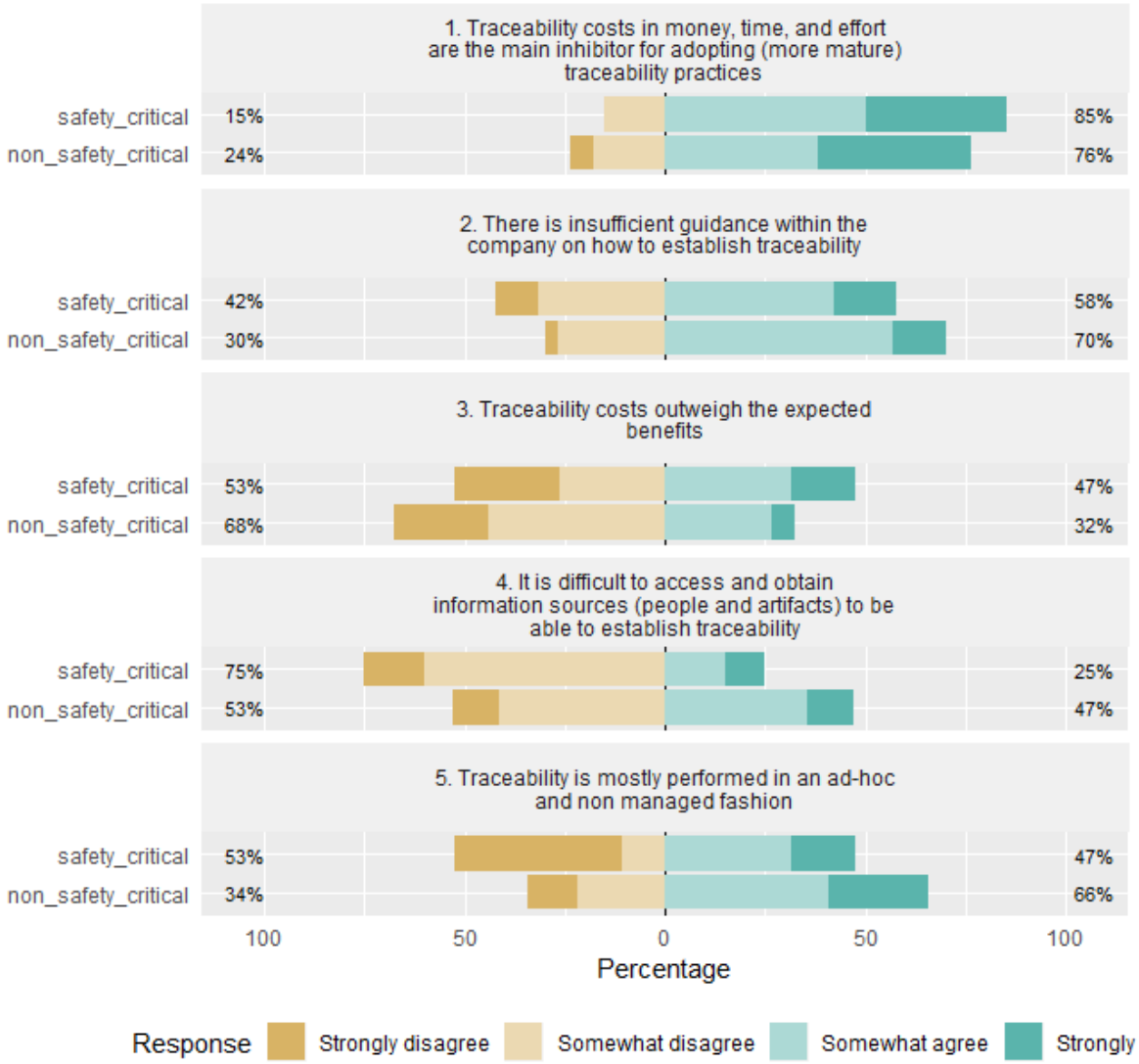


Figure 43: *Divergent bar chart of overall sentiment regarding management current statements between safety critical and non-safety critical*

1. Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices

    a. 85% of our participants that work on a safety critical project agree with the statement (disagree 15% < agree 85%, N = 20). With most of them somewhat agreeing, with 50.00%,

followed by strongly agree with 35%, somewhat disagree with 15%, strongly disagree with 0%.

b. 76% of our participants that follow a non-safety critical project agree with the statement (disagree 24% < 76%, N = 34). With both somewhat agree and strongly agree having 38.24%, followed by somewhat disagree with 17.65%, strongly disagree with 5.88%.

   Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of both groups agree with the statement.

2. There is insufficient guidance within the company on how to establish traceability

a. 58% of our participants that work on a safety critical project agree with the statement (disagree 42% < agree 58%, N = 19). With most of them somewhat agreeing, with 42.11%, followed by somewhat disagree with 31.58%, strongly agree with 15.79%, strongly disagree with 10.53%.

b. 70% of our participants that follow a non-safety critical project agree with the statement (disagree 30% < 70%, N = 30). With most of them somewhat agreeing with 56.67%, followed by somewhat disagree with 26.67%, strongly agree with 13.33%, strongly disagree with 3.33%.

   Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of both groups agree with the statement.

3. Traceability costs outweigh the expected benefits

a. 53% of our participants that work on a safety critical project disagree with the statement (disagree 53% < agree 47%, N = 19). With most of them somewhat agreeing with 31.58%, followed by both strongly disagree and somewhat disagree with both 26.32%, strongly agree with 15.79%.

b. 68% of our participants that follow a non-safety critical project disagree with the statement (disagree 68% < 32%, N = 34). With most of them somewhat disagreeing with 44.12%, followed by somewhat agree with 26.47%, strongly disagree with 23.53%, strongly agree with 5.88%.

   Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of both groups disagree with the statement.

4. It is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability

    a. 75% of our participants that work on a safety critical project disagree with the statement (disagree 75% < agree 25%, N = 20).With most of the somewhat disagreeing with 60%, followed by both strongly disagree and somewhat agree with 15.00%, strongly agree with 10%.

    b. 53% of our participants that follow a non-safety critical project disagree with the statement (disagree 53% < 47%, N = 34). With most of them somewhat disagreeing with 41.18%, followed by somewhat agree with 35.29%, and both strongly disagree and strongly agree with 11.76%.

       Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of both groups disagree with the statement.

5. Traceability is mostly performed in an ad-hoc and non managed fashion

    a. 53% of our participants that work on a safety critical project disagree with the statement (disagree 53% < agree 47%, N = 19). With most of them strongly disagreeing with 42.11%, followed by somewhat agree with 31.58%, strongly agree with 15.79%, somewhat disagree with 10.53%.

    b. 66% of our participants that follow a non-safety critical project agree with the statement (disagree 44% < 64%, N = 32). With most of them somewhat agreeing with 40.63%, followed by strongly agree with 25%, somewhat disagree with 21.88%, strongly disagree with 12.50%.

       Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups, in which most of the participants that typically work on safety critical projects disagree with the statement while the other group agrees with the statement.

7.1.1.2 Social

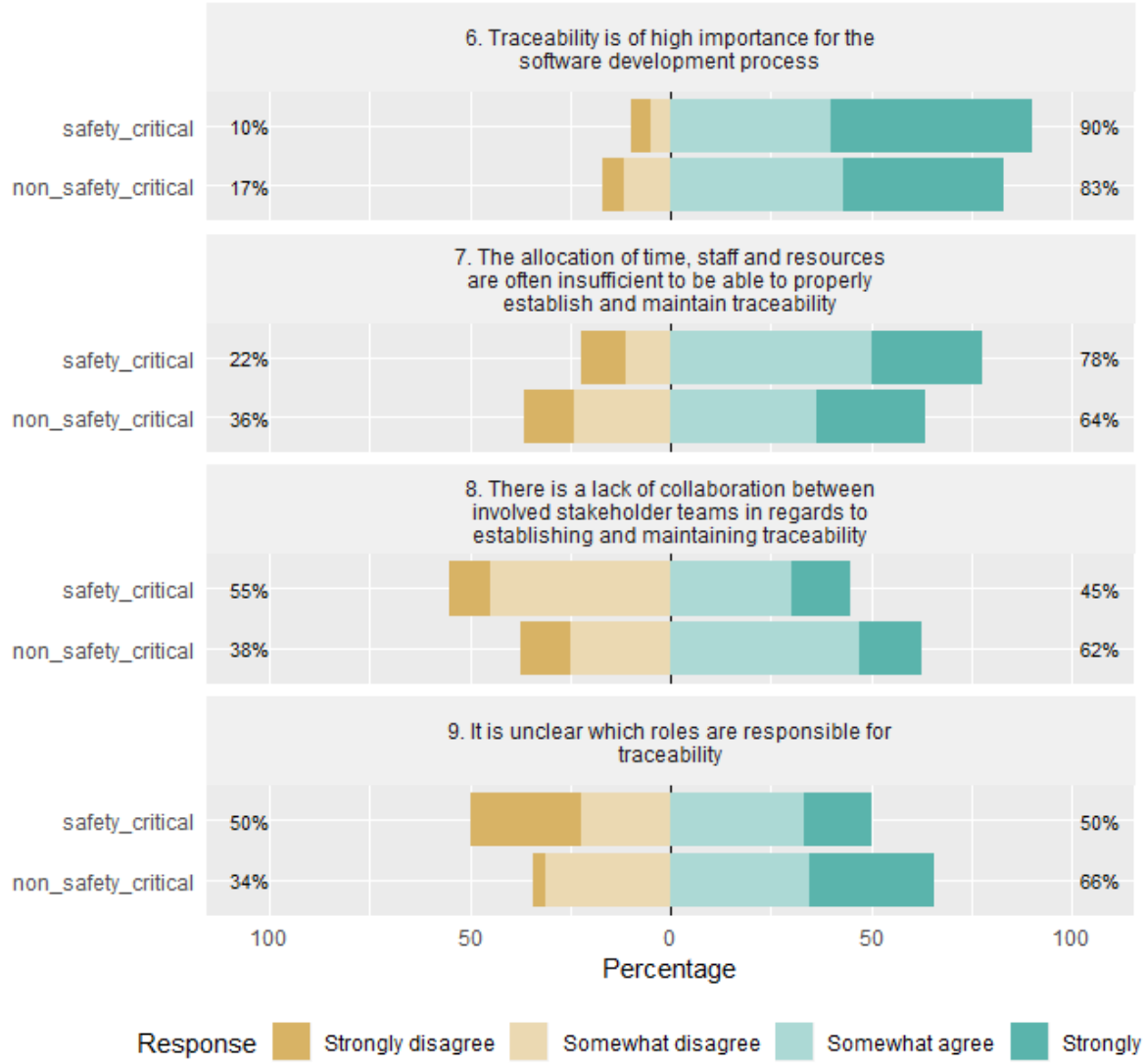


Figure 44: *Divergent bar chart of overall sentiment regarding social current statements between safety critical and non-safety critical*

6. Traceability is of high importance for the software development process

a. 90% of our participants that work on a safety critical project agree with the statement (disagree 10% < agree 90%, N = 20). With most of them strongly agreeing with 50%, followed by somewhat agree with 40%, and both strongly disagree and somewhat disagree with 5%.

b. 83% of our participants that follow a non-safety critical project agree with the statement (disagree 17% < 83%, N = 35). With most of them somewhat agreeing with 42.86%, followed by strongly agree with 40.00%, somewhat disagree with 11.43%, strongly disagree with 5.71%.

   Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of both groups agree with the statement.

7. The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability

    a. 78% of our participants that typically work on a safety critical projects agree with the statement (disagree 10% < agree 78%, N = 18). With most of them somewhat agreeing with 50%, followed by strongly agree with 27.27%, and both strongly disagree and somewhat disagree with 11.11%.

    b. 64% of our participants that typically work on a non-safety critical projects agree with the statement (disagree 36% < 64%, N = 33). With most of them somewhat agreeing with 36.36%, followed by strongly agree with 27.27%, somewhat disagree with 24.24%, strongly disagree with 12.12%.

       Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of both groups agree with the statement.

8. There is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability

    a. 55% of our participants that typically work on a safety critical projects disagree with the statement (disagree 55% < agree 45%, N = 20). With most of them somewhat disagreeing with 45.00%, followed by somewhat agree with 30.00%, strongly agree with 15.00%, strongly disagree with 10.00%.

    b. 62% of our participants that typically work on a non-safety critical projects agree with the statement (disagree 38% < 62%, N = 32). With most of them somewhat agreeing with 46.88%, followed by somewhat disagree with 25.00%, strongly agree with 15.63%, strongly disagree with 12.50%.

       Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups, in which most of the participants that typically work on safety critical projects disagree with the statement while the other group agrees with the statement.

9. It is unclear which roles are responsible for traceability

    a. 50% of our participants that typically work on a safety critical projects disagree and agree with the statement (disagree 50% < agree 50%, N = 18). With most of them somewhat agreeing with 33.33%, followed by strongly disagree with 27.78%, somewhat disagree with 22.22, strongly agree with 16.67%.

    b. 66% of our participants that typically work on a non-safety critical projects agree with the statement (disagree 34% < 66%, N = 32). With most of them somewhat agreeing with 34.38%, followed by both somewhat disagree and strongly agree with 31.25%, strongly disagree with 3.13%.

       Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups, in which most of the participants that typically work on

safety critical projects are equally split over the statement while the other group agrees with the statement.

7.1.1.3 Technical

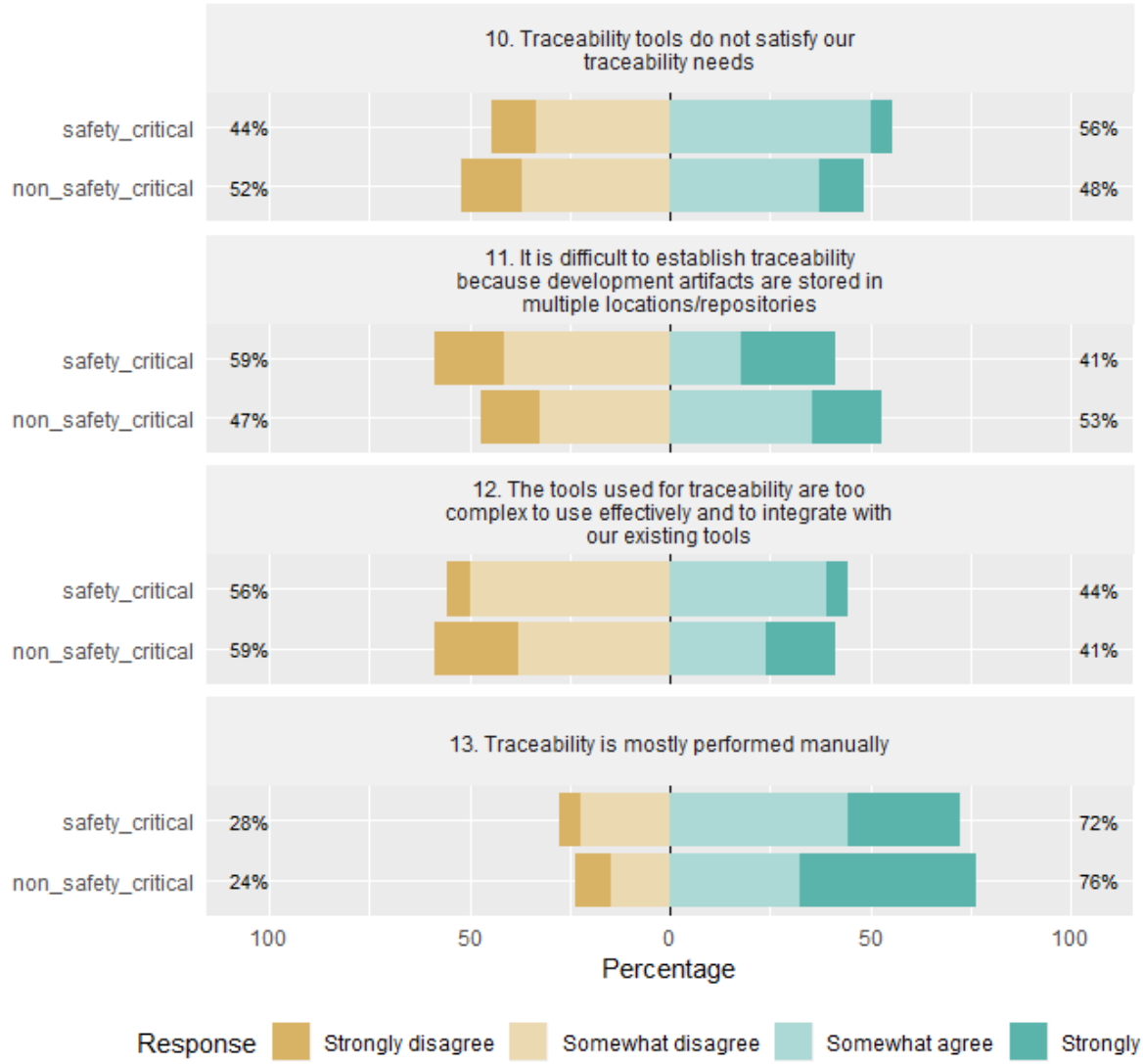


Figure 45: *Divergent bar chart of overall sentiment regarding technical current statements between safety critical and non-safety critical*

10. Traceability tools do not satisfy our needs

a. 56% of our participants that typically work on a safety critical projects agree with the statement (disagree 44% < agree 56%, N = 18). With most of them somewhat agreeing with 50.00%, followed by somewhat disagree with 33.33%, strongly disagree with 11.11%, strongly agree with 5.56%.

b. 52% of our participants that typically work on a non-safety critical projects disagree with the statement (disagree 52% < 48%, N = 27). With both somewhat disagree and somewhat agree at 37.04%, followed by strongly disagree with 14.81%, strongly agree with 11.11%.

Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups, in which most of the participants that typically work on safety critical projects agree with the statement while the other group disagrees with the statement.

11. It is difficult to establish traceability because development artifacts are stored in multiple locations/repositories

    a. 59% of our participants that typically work on a safety critical projects disagree with the statement (disagree 59% < agree 41%, N = 17). With most of them somewhat disagreeing with 41.18%, followed by strongly agree with 23.53%, and both strongly disagree and somewhat disagree with 17.65%.

    b. 53% of our participants that typically work on a non-safety critical projects agree with the statement (disagree 47% < 53%, N = 34).With most of them somewhat agreeing with 35.29%, followed by somewhat disagree with 32.35%, strongly agree with 17.65%, strongly disagree with 14.71%.

       Based on these results, there is a difference in the overall sentiment regarding the current situation between the groups, in which most of the participants that typically work on safety critical projects disagree with the statement while the other group agrees with the statement.

12. Tools used for traceability are too complex to effectively use and to integrate with our existing tools

    a. 56% of our participants that typically work on a safety critical projects disagree with the statement (disagree 56% < agree 44%, N = 18). With most of them somewhat disagreeing with 50.00%, followed by somewhat agree with 38.89%, and both strongly disagree and strongly agree with 5.56%.

    b. 59% of our participants that typically work on a non-safety critical projects disagree with the statement (disagree 59% < 41%, N = 29). With most of them somewhat disagreeing with 37.93%, followed by somewhat agree with 24.14%, strongly disagree with 20.69%, strongly agree with 17.24%.

       Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of both groups disagree with the statement.

13. Traceability is mostly performed manually

    a. 72% of our participants that typically work on a safety critical projects agree with the statement (disagree 28% < agree 72%, N = 18). With most of the somewhat agreeing with 44.44%, followed by strongly agree with 27.78%, somewhat disagree with 22.22%, strongly disagree with 5.56%.

    b. 76% of our participants that typically work on a non-safety critical projects agree with the statement (disagree 24% < 76%, N = 34). With most of them strongly agreeing with 44.12%, followed by somewhat agree with 32.35%, somewhat disagree with 14.71%, strongly disagree with 8.82%.

Based on these results, there is no difference in the overall sentiment regarding the current situation between the groups, in which most of the participants of both groups agree with the statement.

Statement		safety	Non safety
Current_1	Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices		
Current_2	There is insufficient guidance within the company on how to establish traceability		
Current_3	Traceability costs outweigh the expected benefits		
Current_4	It is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability		
Current_5	Traceability is mostly performed in an ad-hoc and non managed fashion		
Current_6	Traceability is of high importance for the software development process		
Current_7	The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability		
Current_8	There is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability		
Current_9	It is unclear which roles are responsible for traceability		
Current_10	Traceability tools do not satisfy our traceability needs		
Current_11	It is difficult to establish traceability because development artifacts are stored in multiple locations/repositories		
Current_12	The tools used for traceability are too complex to use effectively and to integrate with our existing tools		
Current_13	Traceability is mostly performed manually		

Table 24: Sentiment summary table safety vs non safety current situation

Based on the divergent bar chart results and the summarization in table 24, the results show no difference between the sentiment of the groups for the following current situation statements:

- Current_1: Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices
- Current_2: There is insufficient guidance within the company on how to establish traceability
- Current_3: Traceability costs outweigh the expected benefits
- Current_4: It is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability
- Current_6: Traceability is of high importance for the software development process
- Current_7: The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability
- Current_13, Traceability is mostly performed manually

The results do show differences regarding the sentiment between the groups for:

- Current_5: Traceability is mostly performed in an ad-hoc and non managed fashion
- Current_8: There is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability
- Current_9: It is unclear which roles are responsible for traceability
- Current_10: Traceability tools do not satisfy our traceability needs

- Current_11: It is difficult to establish traceability because development artifacts are stored in multiple locations/repositories
- Current 12, The tools used for traceability are too complex to use effectively and to integrate with our existing tools

7.1.1.4 Comparing numeric means

Furthermore, when ranking the perceived current situation statements based on their means, as shown in figure 46 and 47, the results show the top three current situation statements the groups most agree on:

- Safety Critical
  - Current_6: Traceability is of high importance for the software development process
  - Current_1: Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices
  - Current_7: The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability
  -
- Non-safety critical
  - Current_6: Traceability is of high importance for the software development process
  - Current_13: Traceability is mostly performed manually
  - Current_1: Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices

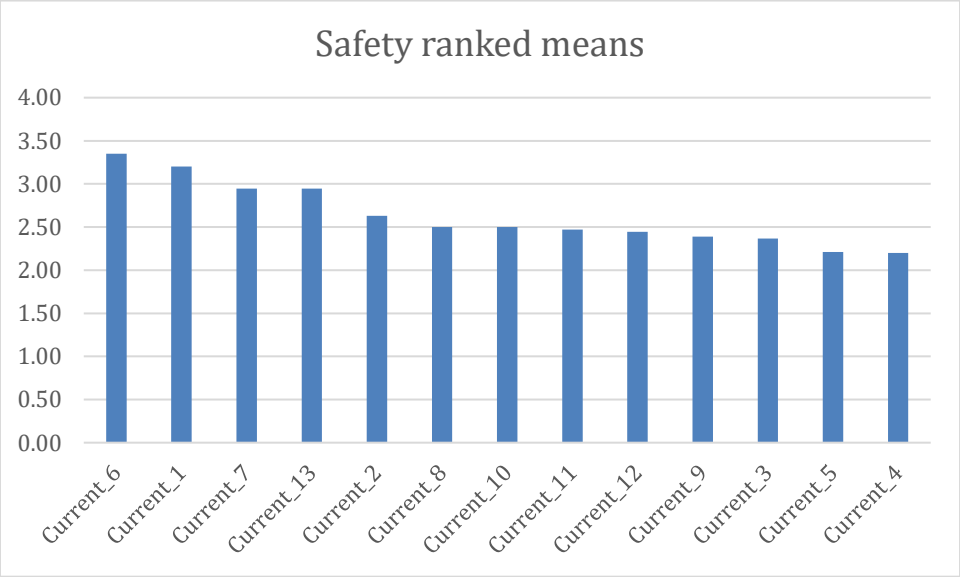


Figure 46: ranked means safety current statements

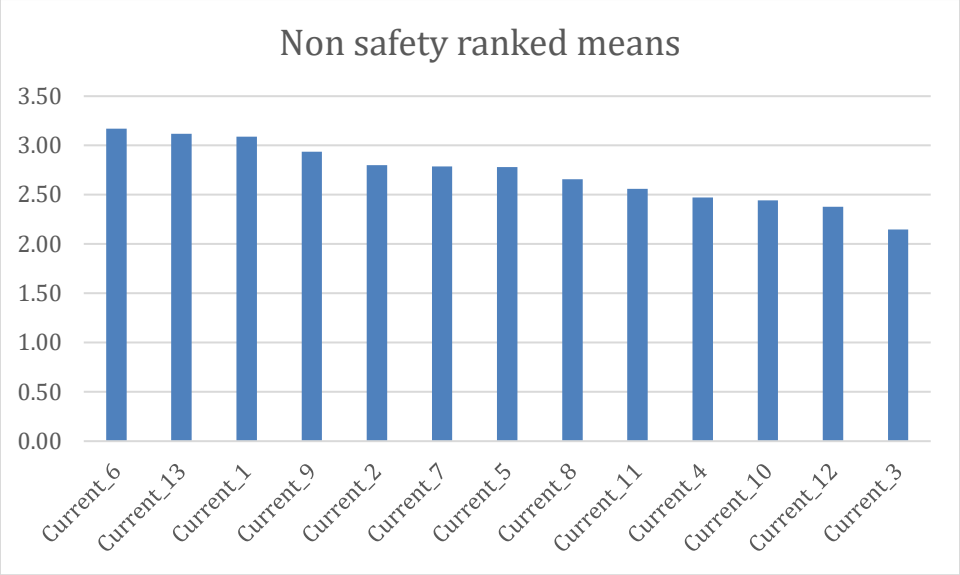


Figure 47: ranked means non-safety current statements

7.1.2 Need

7.1.2.1 Management

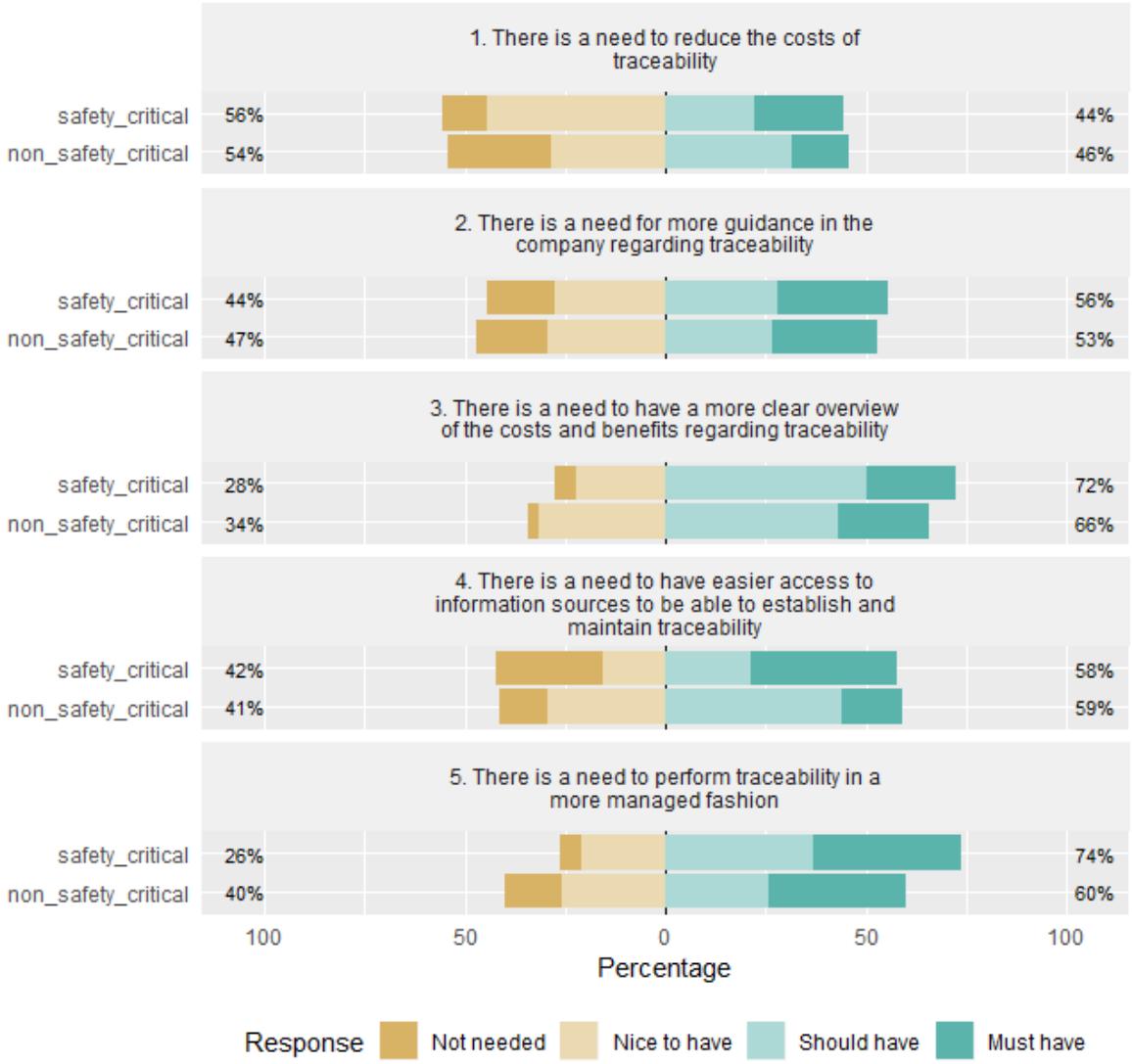


Figure 48: *Divergent bar chart of overall sentiment regarding management need statements between safety critical and non-safety critical*

1. There is a need to reduce the costs of traceability

a. 56% of our participants that typically work on safety critical projects see the need as not important (not important 56% < important 44%, N = 18). With most of them seeing it as a nice to have with 44.44%, followed by both should have and must have with 22.22%, and not needed with 11.11%

b. 54% of our participants that typically work on non-safety critical projects see the need as not important (not important 54% < important 46%, N = 35). With most of them seeing it as a should have with 31.43%, followed by nice to have with 28.57%, not needed with 25.71%, and must have with 14.29%

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of both groups see the need as not important.

2. There is a need for more guidance in the company regarding traceability

a. 56% of our participants that typically work on safety critical projects see the need as important (not important 44% < important 56%, N = 18). With nice to have, should have, and must have at 27.78%, followed by not needed with 16.67%.

b. 53% of our participants that typically work on non-safety critical projects see the need as important (not important 47% < important 53%, N = 34). With most of them seeing it as nice to have with 29.41%, followed by both should have and must have with 26.47%, and not needed with 17.65%

Based on these results, there is a difference in the overall sentiment regarding the need between the groups, in which most of the participants that typically work on safety critical projects see the need as important, while the other group sees the need as not important.

3. There is a need for a more clear overview of the costs and benefits regarding traceability

a. 72% of our participants that typically work on safety critical projects see the need as important (not important 28% < important 72%, N = 18). With most of them seeing it as a should have with 50.00%, followed by both nice to have and must have with 22.22%, and not needed with 5.56%.

b. 66% of our participants that typically work on non-safety critical projects see the need as important (not important 34% < important 66%, N = 35). With most of them seeing it as a should have with 42.86%, followed by nice to have with 31.43%, must have with 22.86%, not needed with 2.86%

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of both groups see the need as important.

4. There is a need to have easier access to information sources to be able to establish and maintain traceability

a. 58% of our participants that typically work on safety critical projects see the need as important (not important 42% < important 58%, N = 19). With most of them seeing it as a must have with 36.84%, followed by not needed with 26.32%, should have with 21.05%, and nice to have with 15.79%.

b. 59% of our participants that typically work on non-safety critical projects see the need as important (not important 41% < important 59%, N = 34). With most of them seeing it as a should have with 44.12%, followed by nice to have with 29.41%, must have with 14.71%, and not needed with 11.76%.

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of both groups see the need as important.

5. There is a need to perform traceability in a more managed fashion

a. 74% of our participants that typically work on safety critical projects see the need as important (not important 26% < important 74%, N = 19). With should have and must have at 36.84%, followed by nice to have with 21.05%, and not needed with 5.26%.

b. 60% of our participants that typically work on non-safety critical projects see the need as important (not important 40% < important 60%, N = 35). With most of them seeing it as a must have with 34.29%, followed by both nice to have and should have with 25.71%, and not needed with 14.29%.

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of both groups see the need as not important.

7.1.2.2 Social

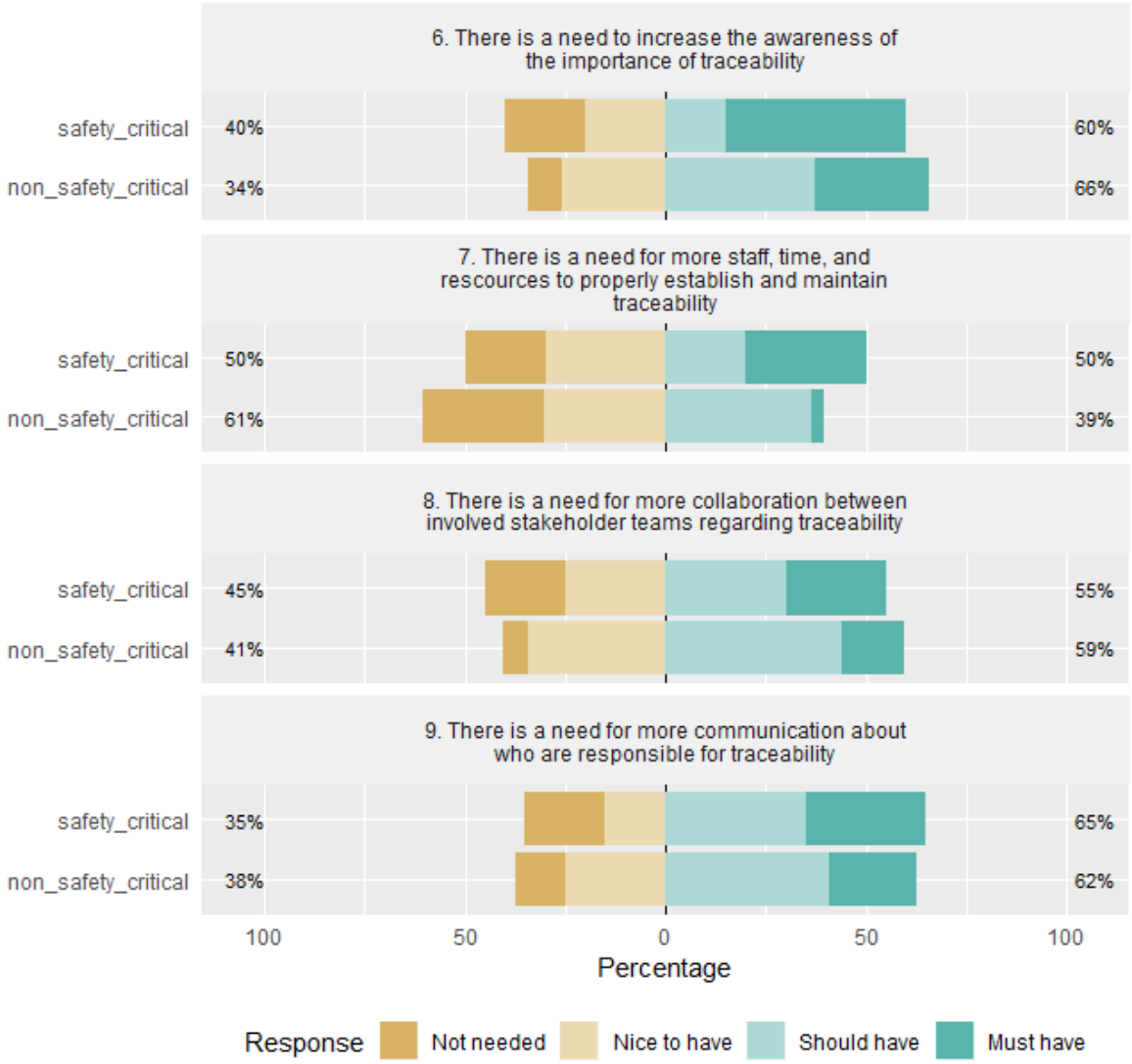


Figure 49: *Divergent bar chart of overall sentiment regarding social need statements between safety critical and non-safety critical*

6. There is a need to increase the awareness of the importance of traceability

a. 60% of our participants that typically work on safety critical projects see the need as important (not important 40% < important 60%, N = 20). With most of them seeing it as a must have with 45.00%, followed by both not needed and nice to have with 20.00%, and should have with 15.00%

b. 66% of our participants that typically work on non-safety critical projects see the need as important (not important 34% < important 66%, N = 35). With most of them seeing it as a should have with 37.14%, followed by must have with 28.57%, nice to have with 25.71%, and not needed with 8.57%.

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of both groups see the need as important.

7. There is a need for more staff, time, and resources to properly establish and maintain traceability

a. 50% of our participants that typically work on safety critical projects see the need as important and 50% see it as not important (not important 50% = important 50%, N = 20). With both nice to have and must have with 30%, followed by both not needed and should have with 20%.

b. 61% of our participants that typically work on non-safety critical projects see the need as not important (not important 61% < important 49%, N = 33). With most of them seeing it as a should have with 36.36%, followed by both not needed and nice to have with 30.30%, and must have with 3.03%.

Based on these results, there is a difference in the overall sentiment regarding the need between the groups, in which most of the participants that typically work on safety critical projects are equally split over the importance of the need, while the other group sees the need as not important.

8. There is a need for more collaboration between involved stakeholder teams regarding traceability

a. 55% of our participants that typically work on safety critical projects see the need as important (not important 45% < important 55%, N = 20). With most of them seeing it as a should have with 30.00%, followed by both nice to have and must have with 25.00%, and not needed with 20.00%

b. 59% of our participants that typically work on non-safety critical projects see the need as important (not important 41% < important 59%, N = 32). With most of them seeing it as a should have with 43.75%, followed by nice to have with 34.38%, must have with 15.63%, and not needed with 6.25%

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of both groups see the need as important.

9. There is a need for more communication about who are responsible for traceability

a. 65% of our participants that typically work on safety critical projects see the need as important (not important 35% < important 65%, N = 20). With most of them seeing it as a should have with 35.00%, followed by must have with 30.00%, not needed with 20.00%, and nice to have with 15.00%

b. 62% of our participants that typically work on non-safety critical projects see the need as important (not important 38% < important 62%, N = 32). With most of them seeing it as a should have with 40.63%, followed by nice to have with 25.00%, must have with 21.88%, and not needed with 12.50%

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of both groups see the need as important.

7.1.2.3 Technical

Figure 50: *Divergent bar chart of overall sentiment regarding technical need statements between safety critical and non-safety critical*

10. There is a need for better traceability tools

a. 50% of our participants that typically work on safety critical projects see the need as important and 50% of the participants see it as not important (not important 50% < important 50%, N = 20). With most of them seeing it as a should have with 30.00%, followed by both not needed and nice to have with 25.00%, and must have with 20.00%.

b. 52% of our participants that typically work on non-safety critical projects see the need as not important (not important 52% < important 48%, N = 33). With most of them seeing it as a nice to have with 33.33%, followed by must have with 27.27%, should have with 21.21%, and not needed with 18.18%.

Based on these results, there is a difference in the overall sentiment regarding the need between the groups, in which most of the participants that typically work on safety critical projects are equally split over the importance of the need, while the other group sees the need as not important.

11. There is a need to have a more centralized development artifact repository to establish traceability more easily

a. 65% of our participants that typically work on safety critical projects see the need as not important (not important 65% < important 35%, N = 20). With most of them seeing it as a nice to have with 35.00%, followed by not needed with 30.00%, should have with 25.00%, and must have with 10.00%

b. 57% of our participants that typically work on non-safety critical projects see the need as not important (not important 57% < important 43%, N = 35). With most of them seeing it as a nice to have with 31.43%, followed by not needed with 25.71%, must have with 22.86%, and should have with 20.00%

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of both groups see the need as not important.

12. There is a need for less complex traceability tools and easier integration with their existing tools

a. 50% of our participants that typically work on safety critical projects see the need as important and 50% see it as not important (not important 50% < important 50%, N = 20). With most of them seeing it as a should have with 35.00%, followed by nice to have with 30.00%, not needed with 20.00%, and must have with 15.00%

b. 55% of our participants that typically work on non-safety critical projects see the need as important (not important 45% < important 55%, N = 33). With most of them seeing it as a nice to have with 36.36%, followed by should have with 33.33%, must have with 21.21%, and not needed with 9.09%

Based on these results, there is a difference in the overall sentiment regarding the need between the groups, in which most of the participants that typically work on safety critical projects are equally split over the importance of the need, while the other group sees the need as important.

13. There is a need for more traceability automation

a. 60% of our participants that typically work on safety critical projects see the need as important (not important 40% < important 60%, N = 20). With most of them seeing it as a must have with 40.00%, followed by not needed, nice to have, and should have with 20.00%.

b. 57% of our participants that typically work on non-safety critical projects see the need as important (not important 43% < important 57%, N = 35). With most of them seeing it as a

nice to have with 40.00%, followed by both should have and must have with 28.57%, and not needed with 2.86%.

Based on these results, there is no difference in the overall sentiment regarding the need between the groups, in which most of the participants of both groups see the need as important.

Statement		safety	Non safety
Need_1	There is a need to reduce the costs of traceability		
Need_2	There is a need for more guidance in the company regarding traceability		
Need_3	There is a need to have a more clear overview of the costs and benefits regarding traceability		
Need_4	There is a need to have easier access to information sources to be able to establish and maintain traceability		
Need_5	There is a need to perform traceability in a more managed fashion		
Need_6	There is a need to increase the awareness of the importance of traceability		
Need_7	There is a need for more staff, time, and resources to properly establish and maintain traceability		
Need_8	There is a need for more collaboration between involved stakeholder teams regarding traceability		
Need_9	There is a need for more communication about who are responsible for traceability		
Need_10	There is a need for better traceability tools		
Need_11	There is a need for a more centralized development artifact repository to establish traceability more easily		
Need_12	There is a need for less complex traceability tools and easier integration with our existing tools		
Need_13	There is a need for more traceability automation		

Table 25: Sentiment summary table: Safety critical projects vs non safety critical project needs. Yellow denotes that most of the participants of a specific group do not see the need of high importance. Blue denotes that most of the participants see the need of high importance and priority. White denotes that the sentiment of the participants are split evenly.

Based on the divergent bar chart results and the summarization in table 25, the results show no difference between the sentiment of the groups for the following needs:

- Need_1: There is a need to reduce the costs of traceability.
- Need_2: There is a need for more guidance in the company regarding traceability
- Need_3: There is a need for a more clear overview of the costs and benefits regarding traceability
- Need_4: There is a need to have easier access to information sources to be able to establish and maintain traceability
- Need_5: There is a need to perform traceability in a more managed fashion
- Need_6: There is a need to increase the awareness of the importance of traceability
- Need_8, There is a need for more collaboration between involved stakeholder teams regarding traceability
- Need_9, There is a need for more communication about who are responsible for traceability
- Need_11: There is a need for a more centralized development artifact repository to establish traceability more easily
- Need_13. There is a need for more traceability automation

The results do show differences regarding the sentiment between the groups for:

- Need_7: There is a need for more staff, time, and resources to properly establish and maintain traceability
- Need_10: There is a need for better traceability tools
- Need_12: There is a need for less complex traceability tools and easier integration with our existing tools

7.1.2.4 Comparing numeric means

Furthermore, when looking at the ranked needs based on their means, as shown in figure 51 and 52, the results show the top three highest priority or most valued needs per group:

- Safety critical projects
  - Need_5: *There is a need to perform traceability in a more managed fashion*
  - Need_3: *There is a need for a more clear overview of the costs and benefits regarding traceability*
  - Need_6: *There is a need to increase the awareness of the importance of traceability*
  -
- Non safety critical projects
  - Need_3: *There is a need for a more clear overview of the costs and benefits regarding traceability*
  - Need_6: *There is a need to increase the awareness of the importance of traceability*
  - Need_13: *There is a need for more traceability automation*

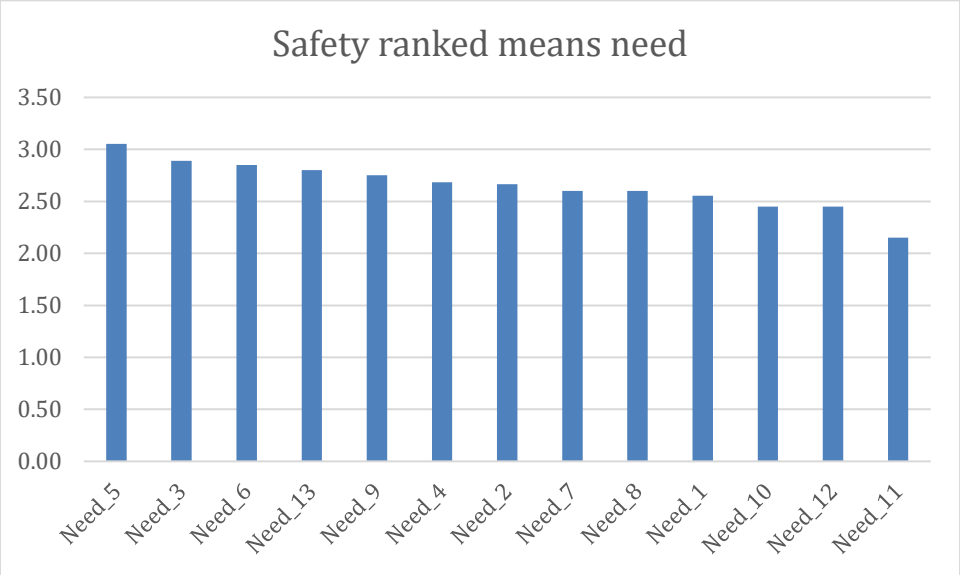


Figure 51: ranked means safety needs

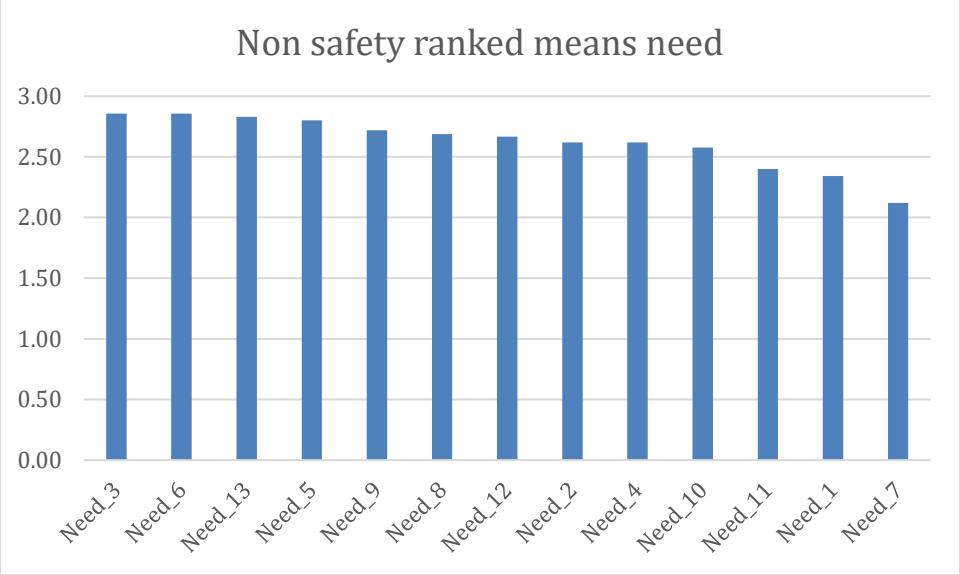


Figure 52: ranked means non-safety needs

7.2 Questionnaire: data analysis

7.2.1 Unpaired Two-Samples Wilcoxon test: current situation

Recall, as described in section 4.1, the null and alternative hypotheses for the perceived current situation between safety critical and non-safety critical projects:

RQ5.1	*"Is the current software traceability situation affected by the type of software project?"*
H0	*The current software traceability situation is not affected by the type of software project*
H1	*The current software traceability situation is affected by the type of software project*

Based on the results as shown in table 26, it is not possible to reject the null hypothesis. Indicating that there is no quantitative significant difference (a=0.05) regarding the perceived current situation between safety critical and non-safety critical projects according to the Wilcoxon test.

Statement	Effect Size	p.value
Current_1	0.11	0.817
Current_2	-0.17	0.509
Current_3	0.22	0.460
Current_4	-0.27	0.207
Current_5	-0.57	0.090
Current_6	0.18	0.418
Current_7	0.16	0.575
Current_8	-0.16	0.450
Current_9	-0.55	0.088
Current_10	0.06	0.786
Current_11	-0.09	0.715

Current_12	0.07	0.728
Current_13	-0.17	0.397

Table 26: Wilcox analysis safety vs non safety current

Therefore, the conclusion is that there are no significant differences in regarding the Likert response distribution between safety critical and non-safety critical projects groups regarding the perceived current situation of software traceability.

7.2.2 Unpaired Two-Samples Wilcoxon test: need

Recall, as described in section 4.1, the null and alternative hypotheses for the perceived value of software traceability needs between safety critical and non-safety critical projects:

RQ5.2	*"Are the software traceability needs affected by the type of software project?"*
H0	*The software traceability needs are not affected by the type of software project*
H1	*The software traceability needs are affected by the type of software project*

Based on the results as shown in table 27, it is not possible to reject the null hypothesis. Indicating that there is no quantitative significant difference (a=0.05) regarding the perceived value and priority of the needs between safety critical and non-safety critical projects according to the Wilcoxon test.

statement	effect_size	p.value
Need_1	0.21	0.513
Need_2	0.05	0.881
Need_3	0.03	0.833
Need_4	0.07	0.699
Need_5	0.25	0.449
Need_6	-0.01	0.841
Need_7	0.48	0.134
Need_8	-0.09	0.836
Need_9	0.03	0.813
Need_10	-0.13	0.704
Need_11	-0.25	0.451
Need_12	-0.22	0.471
Need_13	-0.03	0.949

Table 27: Wilcox analysis safety vs non safety needs

Therefore, the conclusion is that there are no significant differences in regarding the Likert response distribution between safety critical and non-safety critical projects groups regarding the perceived value and priority of the needs of software traceability.

8. Expert interview results

The goal of the interviews is to enrich the quantitative results and to get a glimpse of the rationale behind the answers of the interviewed participants. It also helps us understand the context that they are in and potential factors that influence the needs of the participants.

As can be seen in table 28, there were not enough participants to sufficiently cover and represent every possible group. Therefore, considering these participants have a varied background compared to each other, the interview results will focus on highlighting any commonalities between the rationale of these participants and any notable differences on in terms of their current situation and their needs.

Furthermore, the summarized interview results are structured according to the topics of the questionnaire. The difference compared to the questionnaire however is that the current and need statements of each topic were discussed in tandem so these will be described in tandem as well where applicable.

Section 9 discusses the results of both the quantitative results and qualitative results and discusses the threats of this study and their impacts on the overall results.

8.1 Participant demographics

	Work experience	Current function	Company Size	Development method	Safety critical	Highly regulated	Use	Reason	Current Industry
P1	20 years	Project manager	Large	Traditional	Yes	Yes	Always	because of mandate (regulations/ISO). on request of customer. for the expected benefits.	Automotive
P2	9 months	Product owner	Small	Agile	No	No	Sometimes	for the expected benefits	Retail software provider
P3	1 year +	Product owner	Small	Agile	No	Yes	Sometimes	-	Financial
P4	6 Months	Requirements analyst	Large	Mixed	No	Yes	Always	because of mandate (regulations/ISO) for the expected benefits	Asset management/real estate/mortgage
P5	5 y 6m	Software developer	Large	Agile	No	No	Sometimes	because of mandate (regulations/ISO)	Provider of web application development software
P6	18 years	Software developer	Large	Agile	No	No	Never		Computer software
P7	3 years	Software developer	Medium	Agile	No	No	Sometimes	for the expected benefits	Video games (educational)
P8	9 years	Product owner	Large	Mixed	Yes	No	Sometimes	for the expected benefits	Logistics

Table 28: Interview participant demographics

Traceability view, situation, and reasons for use

One of the questions that is asked during the interview is what the participants think about when talking about traceability. If they mainly think about requirements artifacts and or other traceability related artifacts and which traces are mainly used in their current company. In addition, we also asked the participants to elaborate the reasons of use of traceability. Finally, we asked at the end of the interviews in which phases traceability is most important for them, in which we showed a general SDLC cycle as shown in figure 53. Even though this cycle might not be fully representable, most participants indicated that it was very close to their actual development phases.

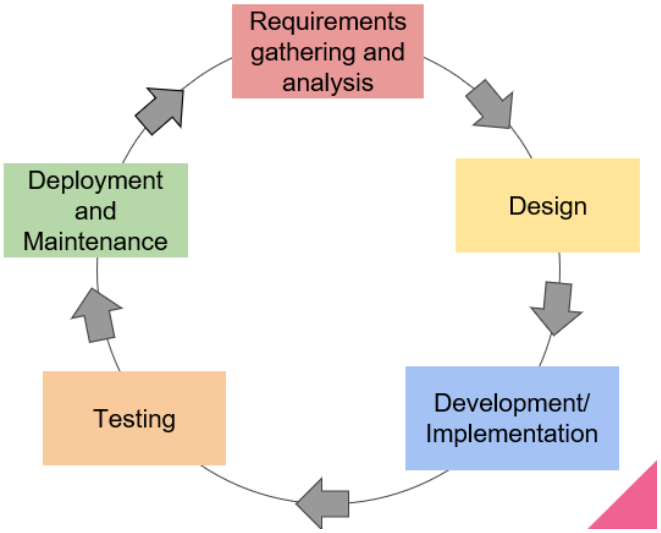


Figure 53: General SDLC

In the case of P1 who works as a project manager at a large automotive company, the participant mentioned traces between artifacts that were currently used in P1s current company. These were traces between artifacts such as such as requirements and test cases, requirements and parameter modules, list of parameter modules and MATLAB modules, diagnostic services to be implemented and the corresponding requirements or parameters. The participant mentioned that compared to literature, that they stopped tracing between requirements themselves. Reason given for this is that they found it difficult to align the traces made by multiple people since multiple people will create different traces. And since they could not rely on those traces, they would not bring them much benefit. An example as given by P1:

> *"It is not so easy to come to a very clear definition. So, in one case my colleague wouldn't draw a trace and I would do it or vice versa. So, nobody could rely on the traces that the other person had made. And so well, I told my colleagues, well traceability has to survive the student test. Which means I must be able to pick a student from the street with no idea of the subject. But I can sit him in front of the computer and say well please check the traces if they make sense. And he is able to check, well that the test case has something to do with the requirements. Even without being an expert I can say it is about opening the rear door or the requirements about opening the rear door makes sense. But to say that well those requirements are in a contradiction or support each other or whatever. No student from the street would be able to do so. So that, this kind of traceability wouldn't survive the student test, so we just stopped this kind of traceability. There was no real benefit out of it."*

When asked to elaborate the reasons of using traceability, P1 mentioned that it is in part because of regulations that apply in their area. As exemplified by P1:

"The regulations that apply here, especially the iso 2062062. So that is relevant. Well you are not forced to adhere to this standard. This is only relevant in case of product liability cases. So, I can release vehicle without having any traceability. That is not demanded by the regulations but in the case there is a severe accident. So I have a product liability issue then of course there would be the question, well why didn't you adhere to the standards for developing safety critical systems and they demand of course traceability, so it is not necessary in the first step. But in the second step, so in case of, like a health insurance. You say well it makes sense so that, that is one reason for using traceability."

Additionally, another big reason for using traceability as P1 described was to mainly just make life easier and to be able to make it easier to handle the large numbers of requirements that they receive on a periodic basis and to be able to analyze the impact of changes or requests and to check which tests cases have to be rechecked.

Finally, when asked in which phases of the SDLC traceability was most important for the participant, as seen in figure 53, the participant stated the following:

"If I have the viewpoint of the OEM car manufacturer. Then it is mainly, the main importance I would say in the testing phase. Because this is one of our core activities, besides the specification and in the specification itself we do not use traceability intensively. If I look at our suppliers, and usually I go in the term of performing audits to our suppliers and have a look at their traceability chain starting from the requirements down to the testing results. And here of course, it is especially essential in the, well, from design down to testing. "

P2 who works as a product owner at a small sized company that provides retail software for many customers, also mentioned the traces and artifacts that were mainly used in P2s company. In the case of P2s company, there are two layers regarding tracing, which are between the business requirements for the business stakeholders and user stories for the developers, in which P2 is responsible for translating between the two. In addition, they also have traces from user stories to tests cases, and traces between the documentation in different tools such as Jira and Confluence. Furthermore, P2 mentioned that they also have a weak trace between pieces of code and software features:

"Then there is a trace link between the software and the requirements. Because when we deliver a piece of code we mention which feature it solves. So there is a trace link there. Even though it is not that strong."

When asked to elaborate on the reasons of using traceability, P2 mentioned that one of the reasons traceability is used is to figure out solutions to problems, in which P2 gave the following example:

"The key thing that we notice here is that we have many different customers. … We have over ten million devices running every day and every retailer is different. So, this means that every retailer encounters different problems which are caused by different things. Mainly for our support department if they encounter a problem, if a customer reports a problem, they can dive into Jira and they can search for things and they can maybe find another customer that has reported the same problem and they can click on to see if the dev task was already done or can find some documentation. Our software is heavily built on parameters as well. So, you have over a thousand parameters. Maybe for a problem the customer has there might already be a parameter that fixes it. But yeah this is twenty-year-old legacy software, and we are a quickly growing company so not everyone has the knowledge yet. And we have a couple of those knowledge oracles running around in the company, but they are not available to answer all your questions."

Another reason that the participant mentioned is that it provides their higher ups an overview of what is going on. In which e.g. project managers can all the way click down from deliveries to tasks, to developers' tasks, to implementation tasks for consultants. The biggest advantage that P2 mentioned that traceability brings them is knowledge, as in having an overview and being able to find things and link those to for example even 15 year old tickets, which helps them with maintaining their software and creating new features for all their different customers.

When asked in which phases traceability is most important, P2 stated the following:

" Definitely *in deployment and maintenance. Yeah that is where you find out and stuff is not working and you want to know why. And then you can find out if you have missed something or if you need to set a parameter, if you can fix it with a workaround that somebody else already did. So that is the main part. Of course, the development and implementation would be the next part, but, also in the company I see that the biggest benefit is with the operations and the support department."*

P3, who works as a product owner in a small startup in the financial industry, described traceability from a higher level of abstraction and mainly focused on pre requirements traceability. When talking about traceability P3 mentions that it is mainly about decomposing an idea or problem to solve into smaller requirements which are linked to a common bigger goal and being able to trace the origin of the idea or problem which for example could be customers themselves or internal stakeholders. Besides P3s description of post requirements traceability P3 was also aware that this was not the only perspective on traceability and in which the participant gave examples of traceability between artifacts such as design documents and technical approach, features and requirements, and development tickets and code.

"It is almost like this big unclear problem or request, kind of moves forward and is like broken down to smaller problems. And as the process moves on it also becomes a bit more clear for everyone of what we actually need. And then, it is almost like this big problem becomes also like smaller requirements. But all those smaller requirements, link somehow to a big goal."

When asked to explain about the reasons of using traceability the participant mentioned that it is mainly used for tracking purposes and to be able to relate the goals and or problems with the user stories and the user stories to the tickets and sprints, which is mainly done by a combination of Trello cards and labels, GitHub, and other small tools they use. As described by P3, when they get the requirements, they start thinking around features and how to solve a certain problem. P3 mentioned that it is almost natural that their conversations start to shift from the problems to the features they are going to develop, towards how they are going to implement it.

When asked in which phases traceability is most important, P3 stated the following:

" *So yeah, looking at this cycle. Like in my personal case what is more important, like this link between requirements gathering and the design, and the implementation. But I think, looking at the whole cycle, I can also think of other examples. Like for instance, maybe that is more on the technical side, like when you are developing a new feature, and you kind of need to touch parts of the system. That sometimes, when you start testing, and figure out that a change on a certain feature actually affected those other three features because they use the same backend or whatever. So maybe being able to anticipate that more easily, would also be helpful. Because I think most of the cases, most of the times, we try to anticipate it by literally having the engineers think about. Okay, where is this part of the code used? But they don't really check it, it is more thinking about it. And then, also, I think when we start testing, I think it would also be helpful If you kind of know. Okay, we are implementing this new feature that touches upon this part of the system. So, we should also put a bit of more effort*

in testing those other features, 'c' and 'd' because they use the same resources and because we changed it so they might be affected."

P4, who works as a requirements/business analyst at a large company in the real estate and mortgage company, described that traceability is about the origin of requirements until its implementation. However, P4 added that the origin and creation of requirements was not done by the internal IT where P4 is currently situated. What is important to note in P4s case is that their IT department acts more as a governor for changes and keeps an overview and manages the incoming changes from other business units. These changes are then analyzed and when approved they are relayed to external parties that implement the changes to the systems and software. So the moment a new requirements come in, they perform an analysis to see what the impact is of the change which is through a standard workflow that includes traces between artifacts such as the component, a functional design, a risk analysis, test plans, and acceptance tests, which all eventually leads to the release of a change. This workflow is performed as part of the service platform they use, named ServiceNow that is an enterprise software that digitalizes workflows in which they have set it up that change requests are submitted via this platform. As such they act more like an external department that mainly focuses on managing changes for multiple development cycles in different business units and they do not or are not fully part of the software development cycles that the other business units go through.

"*See it as, IT manages and directs the overview of all the changes that happen. But the business units are responsible for the change. So, the moment there is new legislation, the business units have to submit a project or a change in our platform. And then we take it and we check what the effect is of that change for the IT.*"

Furthermore, P4 mentioned that they do not specifically have a product backlog in common terms but that they do have a product backlogs which keep track of smaller parts in which the requirements are present and clear for the minimal viable product and the included priorities. However, they do not keep track yet of when specific parts have to be finished over longer periods of time but are mainly focused on the present. As explained by P4:

"And, eventually, now it runs the minimal viable product but there still needs to be a lot of development done on it. So in that regard we do have a product backlog, but it is more focused on specific parts, and not on the general whole. Because I see a product backlog also as a predictor, and that predicting we do not do. So in the sense of when do we want it. Our product backlog right now is more about the, what is currently present and which ones have the highest priority to be developed. … unless we have a lot of changes then we will have something for over a few months but in principle we just look at what has to be developed first and then agreements are made based on that. "

When asked about the reasons of using traceability the participant mentioned that it is in part for audits in which they have to be able to show that they followed the standard change process. In addition to the audits, the participant also described that it is used for tracking purposes in which the biggest advantage is to be able to keep an overview and to check in hindsight how things were done. As P4 described:

"…, but also to keep track ourselves if we want to know what happened a few months back on that area, then we have to track it down. We work with a lot of outsourcing partners, so we have to be able to keep an overview. In the end, it is our software, but it has to be customized to a lot of external suppliers. "

Lastly, P4 also mentioned that it helps them invoice the costs that have been made based on the tickets, and to check if these tickets and or invoice requests are valid.

When asked in which phases traceability is most important, P4 stated the following:

"*Yeah mainly from analysis and design. Just taking a look. Well you also need it at the moment you develop something. And there are dependencies. Then you will need to bring those dependencies in 'card' view. And in the moment two people are writing on the same thing, then things will of course go wrong. In that regard, I think deployment and maintenance are actually the least important. But during the design you already have to take it into account. And with development and implementation it is also important because you need to know when something falls, why does it fall? And with testing it is, why does it not work and are there maybe other causes next to what they developed? But I think for requirements gathering it is very interesting to see what is about to come in. But we do not have that yet. So maybe deployment and maintenance the least interesting, except if suddenly at our production environment things fall. But in essence that should have been tested well already before it goes live. Yeah. And in the end, maintenance I do think again that if maintenance needs to be done, but not really for putting it live. So, I do think for deployment and maintenance, that I do agree with maintenance."*

P5, who works as a software developer in a large company that provides web application development software, mentioned that for their work they mainly keep track of stories and bugs via Jira. Furthermore, each backlog item has a code which they use to relate it to corresponding changes and commits, or tickets so to speak. However, the participant also noted that this is not strictly enforced but that it more the way of working that is encouraged throughout their online platform, in which the platform provides traceability support by allowing the developers to append codes and messages to commits. As described by P5:

"You know you make your changes and then you commit, it is not strictly enforced but it is sort of the way of working that is encouraged throughout the online platform, is you prepend your commit message with the name of the ticket that you're working on. So each commit message will be like, #AS-450: … changes that I did in this commit."

When asked about the reason of using traceability, the participant mentioned that it is mainly because of the audits that they have. As P5 mentioned:

"But, so the reason we have, I feel like the major reason we have all the traceability that we have, each change needs to be linked to a story, and each story needs to say that it has been peer reviewed and tested, and all these checks. The reason we have that is for the audit. "

In which the participant added that it is likely so that they can market themselves as having good quality. As P5 stated:

"Yeah, and I guess the reason we have the audit is then, we can then market ourselves as we have proof that we have/are high quality, see this third party that says we have good quality checks."

Interestingly however, P5 was unsure if doing traceability really improved the overall quality, since it is difficult to prove. As P5 stated:

"I don't know whether doing that increased overall quality. It is hard to say. I do know of one case where some people forgot to include the story number in their commits. And then, one or two

developers had to stay late right before a big release of trying to map them all correctly. But I think that was for an audit. But, again, it is prompted by the audit."

When asked in which phases traceability is most important, P5 stated the following:

"*I see in deployment and maintenance, like. The one benefit I see in traceability is like if there is a bug if determining, oh is this something that we overlooked or is this something, we made a code change, and it introduced this bug. And so, you discover those bugs when they are deployed and then maybe you get like support tickets that say hey, this is not working. So having traceability, between, yeah when it is deployed. What the functionality that is seen on your product and then how it was in development. So, I say between the green and the blue, having traceability between those two is good. And also, probably testing as well, because it is in that loop of, if you find a bug and try to fix it like is this something that was introduced in the development cycle or did we skip it. If we skipped it then in testing, make sure that behavior is captured in testing and then actually fixed in there."*

In case of P6, who works as a software developer in a large company providing computer software, the participant mentioned that based on the questionnaire, the participant filled it in from the perspective of source code to requirements, in which source code could be running in production or tests for the code in production. However, for P6s job it is also about the deliverables like binaries to source code and the steps before. Basically, from what is deployed somewhere to why it was done. As stated by P6:

"*Yeah. So, I think when I answered the questionnaire. I was thinking from going back from source code to requirements. And source code could either be, source code running in production or tests for the code in production.* "

It is important to note that because of this, the answers of P6 only reflected on how useful or beneficial traceability from source code to requirements would be for P6s role without taking into account other forms of traceability which were eventually also discussed with P6 and in which P6 had different opinions about compared to the perspective from source code to requirements.

When asked about the reasons P6 mentioned that traceability between code to requirements was not used but that going from code to push/pull requests, to commit are done on a weekly basis. As P6 mentioned:

"Yeah code to requirements. So I can't really remember doing that, ever, in the last 20 years. But going from code to push/pull requests, to commit, that is done weekly or something like that."

Furthermore, going from bug to pull requests was also often performed. Main reason for using this type of traceability is so that they can understand why something was messed up.

When asked in which phases traceability is most important, P6 stated the following:

"So, where I use it is from deployment and maintenance to development. So, we don't have the testing and deployment. So, the testing phase here for us is kind of split. We have some testing done in development and implementation and we have some testing done when we roll it out or call it out in deployment and maintenance. So, it is divided between those two. The blue and the green. And then going from the green to the blue. That is where we use it.

P7, who works as a software developer in a medium sized company that provides educational video games, described traceability also based on the setup of their current company, which is a startup. P7 mentioned that as a startup that they usually did not need any extensive or big traceability solutions and that they mainly used some small and common stuff such as git and source tree to keep track of everything along with a platform called Trello. The participant also elaborated on the traces that are available based on their setup, which are basically the basic post requirements traceability traces between requirements, test cases, and code, which are supported by the smaller tools. As P7 stated:

"So in terms of traceability, I mean, as a startup we usually for tracing we haven't used anything really big. It is mainly git with, we currently are using source tree to keep track of everything along with small platform called Trello. So Trello, git, nothing too complicated actually. And that is how we usually, yeah. "

When asked about the reasons of traceability use P7 mentioned that it is mainly to keep track of everything and to make sure of how they got to certain stages in the development, especially in cases of issues with certain builds. P7 also elaborated on the reason of sometimes using traceability, which is mainly attributed to the startup environment in which the participant is situated. As P7 elaborated:

"So why sometimes, I guess also being a startup. Sometimes, not everything is recorded down. Which is not the best practice to follow. But sometimes things will be discussed. And we'll test it right then and there. And nothing would be noted down in that case."

When asked in which phases traceability is most important, P7 stated the following:

"*In general, we do kind of follow this cycle. Maybe, sometimes switching some things here and there but in general we follow it yeah. Requirements gathering, analyze data, then design, and start development work. The testing, yeah, the thing with our phases. Is usually for example the developer would do three of these phases for example. Like development testing and deployment. We wouldn't have separate teams for them. But in general, yeah, we do follow this cycle as well. …Yeah it would have to most impact in the design. Well during the designing when you come up with ideas etc, etc, the plan to work, would definitely be good to have a record of what was really planned and discussed etc. And deployment. Especially with development work. Just having a really, a record has been done. "*

In case of P8, who works as a product owner at a large company that provides software in the logistics industry, the participant explained that they strived for a complete end to end traceability flow. As described by P8:

"So, with my focus, requirements traceability and requirements tracing is of course, personally, very valuable and very important issue. But if I look to my work, it really starts with high level concepts that I need to trace to more detailed requirements for the concept and high-level architecture which is related to how we are developing software at the moment. And then the final split of user stories for the developers, the code tracing between user stories and code. Then the tracing between user story, requirements, and test case. And the overall feature tracing related to completely the tree back, to the requirements, to the code, to the different versions of the software, to the different sub-services, sub-parts of the software and back to the high-level requirements. So, it is really, for me it is complete, I would say it is a complete flow. "

Even though this description indicates a complete flow, the participant mentioned that not everything was always up to par depending on the setup and situation. As stated by P8:

"But it doesn't mean that all of the, it is in a good mood or place. So, this is what we are talking about as a company, this is what we are heading for but it is available depending on setup, depending on the situation in a different way."

When asked about traceability usage and the reasons P8 mentioned that they sometimes do it mainly because they are not in state yet to have it everywhere and to do it in every situation. As for the reason, it is important and the current goal for them to know which customer is using which piece of software that is related to which requirements, which features, to which bugs. Reason for this goal and wanting this end to end traceability as given by P8 is that they not only work with a fairly long existing software product but so that they can also check for safety critical aspects and also security related aspects which plays an important role in their area.

In addition to that, P8 mentioned that since they work in a very big setup which is also still growing faster than they expected, that it is important to track and keep an overview of what people are doing. As described by P8:

"The other side, that we are working in a very big setup. We have at the moment something between 150 and 200 employees only working with this single product piece. And we need to make sure that whatever one person is stating and writing down in requirement document, this is taken into account. Have taken into account what is refused for development or developed but we need to trace it. If not, we will very easily lose control. "

When asked in which phases traceability is most important, P8 stated the following:

" I think in our environment, it is very important to have, I always call it a full-blown tracing approach in place. But if you ask me to reduce it to a minimum needed setup, then of course between requirements, testing. As testing is somehow presenting as what is in the development. And is making sure that what is developed is in line with the requirements. This is the absolute minimum that we need. Requirements to test. But I would really like to have a full-blown trace support. "

8.2 Current situation and needs

This section describes the common themes and differences between the rationale of the interviewees regarding the statements and the factors influencing their needs. In cases where a similar theme is mentioned the results are summarized and only a few excerpts of participants are given to highlight the theme and or factor.

Costs

1.1	Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices
1.2	There is a need to reduce the costs of traceability

According to P3, the answer on these statements might depend on the type and the state of the tool that is used. As P3 mentioned:

"I think it probably depends a lot on the tools we use. So, the moment we are using Trello, which is like a very flexible tool. But at the same time, it is not specific to product development or something. So that also means that if you want to trace requirements to features, or link things together, it also represents a lot of manual work. And there is always, like you know, you need to be aligned between

different teams, and sometimes like the same initiatives there are different teams working on it. So, it ends up, like there is a lot of effort and time involved."

This seems to be exemplified by P8s quote which indicates that the used tool(s) have a lot of impact regarding answering these statements about traceability costs. As P8 mentioned:

"It is just that, as I mentioned we have different tools in place. And each of them is built as an isolated solution for small sub-part. And nobody ever looked at the complete flow to check can we automate aspects, can we support aspects, and how to optimize this. The current setup is leading to the situation that I, as a product owner, I am writing user stories in Jira, and I am writing related to the same user story, requirements in DOORs, so I have twice the work. And the DOORs document is only used for traceability reason. Nobody will read this document. This document is only in place to trace to the higher levels"

Furthermore, one of the noticeable themes that almost all the interviewees had strong statements and agreements about was that the effort is the main inhibitor compared to money and time for adopting (more mature) traceability practices. Not only is this apparent in the previous quotes but also as P1 stated:

"… so, if I would have additional information and so on. This of course, would cost money, time, effort, it's in the end effort to be spent by the engineer and here I would say they wouldn't see the, at least given the current tool support, there wouldn't be the benefit for this added effort."

Not only do most of the participants think it is a lot of effort, but some even indicate that it sometimes feels like a waste of effort and that it prevents engineers from focusing on the tasks at hand as seen in the quote of P8 and also as stated by P6:

"Yeah, traceability costs in money, time, and effort are the main inhibitor for traceability practices and that I agree in. It's, it's because it is tedious paperwork which we don't use so it is waste."

In contrast to most of the interviewees, P4 and P7 both disagreed with the statement and indicate that costs are not an inhibitor for their environment. The rationale as given by P4

"I think it only delivers money at the end of the trip, so maybe even strongly disagree. Because traceability makes it possible that you can provide better software and that you will lose less time on errors that can happen. And if you have a platform like this, it basically helps you more with the process. Helps with keeping the overview. So I think it is actually really important."

In case of P7 the rationale is mainly based on the fact that P7 is in a startup environment and that in case they need traceability, their main company can provide them with the necessary resources. As stated by P7:

"So as -Company G- does have, as they are a very big company, they can provide us with resources to help. But currently it is not necessary for most of us startups. To be honest. Until we essentially grow bigger."

Based on our interviewees, the effort for the engineer seems to be the main inhibitor for adopting (more mature) traceability practices, in which part of the reasons given is that the effort itself is related to the current tools used and the tool setup in their companies. In addition, the type of environment, e.g. if it is a large mature company, growing company, or a smaller startup influence the perceptions if the costs are an inhibitor or not, and their need to reduce the costs.

Guidance

2.1 There is insufficient guidance within the company on how to establish traceability

2.2 There is a need for more guidance within the company regarding traceability

Multiple participants agreed that there is insufficient guidance regarding traceability. As P7 described:

"So that is the idea that. I placed somewhat agree to insufficient as, once again bringing it back to the startup idea. Is that, when I joined as developer. There weren't really any set rules. On really how to establish traceability. It was more like, here you go, use source tree and use Trello. We didn't really follow a really strict system. So that wasn't really any real guidance. As long as you can keep records, it works. So it was basically that idea really. It is very very, yeah lackluster in terms of that department."

As P7 pointed out, in this participants case it was mainly attributed to lack of knowledge sharing when being onboarded and also being in a fast-growing startup environment. This is further exemplified when looking at P8s rationale, who describes similar themes and also works in a fast growing but larger company. As P8 described:

"We have some tool setups in place to support traceability. Problem is, is that we are using different tools, just from growing companies every area started with a separate tool. There we have for example in place for requirements, DOORs. As a requirement management tool. We have on the other side HP quality Centre as an application life cycle management tool which focus on testing in place and in between Jira as ticker or task management system, and bitbucket. And the problem is that all these single tracing aspects were only done related to one tool. And the chain was not closed as the people were not aware. Someone once set it up for one of the tools, but most of the people working with it didn't get an explanation of why it was needed, didn't get an explanation on how to use it, and they just saw it is a lot of effort, so why should we do it. So, we have just some missing areas between all these single linking aspects. And therefore, there is missing guidance. As I started in the company, I didn't get an introduction into that… I was aware that that is an issue but for most other people starting in the company, nobody is telling you that this is important and how to do it. It is quite challenging and what I see is just some of the tools are in place or some of the tracing setups are in place but not used and nobody knows how to use them."

As can be seen in the case of P8, even though there are setups in place to support traceability, not all the used tools are properly documented and similar to P7 there seems to be a lack of knowledge sharing for new employees when joining their respective companies and or teams. In addition, it should be noted that even though general knowledge sharing might be performed well in all the participants cases, it does not necessarily include knowledge sharing of traceability itself. As P2 described:

"I mean, we do try to transfer knowledge. So, when you start working here you have a training session and everything. And you can always ask questions of course. But that is not related to traceability. That is just general knowledge sharing. "

Furthermore, guidance might also differ per team depending on their best practices. As P5 stated:

"I would say it is knowledge passing, and also it depends on each team. So, I know, for our team and the team I used to work on. It was said how I should do it. And also, when you peer program. But I don't know if other teams do this. "

For the interviewees that disagreed with having insufficient guidance there were several different reasons. For the case of P1, their company already has trainings in place which is also supplemented with general knowledge passing practices. As P1 mentioned:

"Yeah, there are trainings. If you start, you either ask your direct colleagues or you go to internal trainings. And there you have been told how to use the pretty simple traceability. So, it's pretty clear for the engineer on what to do."

In the case of P3, they do not have trainings, but it seems that the knowledge passing in their startup environment is enough as P3 described:

"So, I feel like, again in startups, things move kind of fast. And I don't think there will ever be some sort of training around that. But I feel like, as the company grows, and depending on the people who were there before. Like people, I think, naturally, start adopting some practices. And then, for instance for me, who joined a few months ago. It also took me a while to understand, what were the practices that were in place. Even if it was not like very formal things."

Furthermore, P4 mentioned that it is mainly about learning the tool or system that they are using and that they do not have to think about it anymore. As described by P4:

"Well, we basically have the platform. So, I think, the first thing you have to deal with is this platform. And you just learn how to use it. And in addition to that, this year, we had to deal with the interior design of the platform because we started using it at the start of this year. But it is basically learning it and then use it and then you don't really have to think about it anymore. "

In case of P6, the participant mentions that in their current company they do not really have clear rules to follow but the participant also mentioned that this is an advantage because it makes them move faster. As stated by P6:

"Yeah, there it was rules for what you have to do to be able to check in. So that was part of that. But here we don't have it and I think that makes us moving faster."

Based on this it becomes clear that even though there might be enough knowledge sharing in a company, traceability knowledge and guidance is often overlooked. Furthermore, the differences in the participants situations and their need regarding guidance can also differ per team depending on the way of working. In some cases, it is attributed to the tools used and just learning the tools, which influence the answers.

Return on investment

3.1	Traceability costs outweigh the expected benefits
3.2	There is a need for a more clear overview of the costs and benefits of traceability

In the cases of the interviewees that agreed that costs outweigh the expected benefits in regard to traceability. One common thing that was notable was that in the case of P3 and P5 the benefits were not clear while the costs were more apparent. These participants did note several perceived benefits but mentioned that not all the benefits are known. They still see the costs outweighing the benefits,

which is also the reason that there is a need for a clearer overview of the traceability costs. As stated by P3:

"Because yeah, when I think of traceability I tend to think more of, linking problems to solutions. … So, it could be that there are things that are not on my radar right now. Because I also don't think too much about those other things. I am also not sure of what would be like the extra benefits and costs. So, one benefit that I can think of, is linking a feature request to customer. Once you deliver that, you can go back to the customer and say, hey, remember when you requested that feature, we actually developed that. Or, hey, remember the bug you reported. We actually fixed that. So, I think in those moments they bring very clear benefits. Because you are pleasing your customer basically. But I think, internally, if you link problems to solutions, it probably helps the team have a bit more context on what they are doing and not just like doing code because someone told you to do. But probably there are some other cases at the moment that have benefits that I do not know about."

As described by P5:

"Exactly. What is the expected benefits of traceability. I don't want to have to do busy work just because I am bored. I want to know that this work that I am doing with traceability is going to give some benefit or improvement of quality or transparency in how everything is related to each other. The expected benefits are like minimal, or unknown right now. And costs are much more apparent. So that is why I somewhat agree in that regard. "

In the case of P6, the questionnaire was filled in in respect to code to requirements traceability in which the participant did not see any benefit in it for its role. As described by P6:

"Yeah, that is based on my understanding of this questionnaire that was from code to requirements. So, from code to pull requests or commit or what you want to call it. That is useful and that I use. But to the requirements I have not really seen the use of that and the costs outweigh. So, the benefits for me from the last 18 years have been zero. So, the costs outweigh it."

However, when asked about traceability that they currently do use, the benefits were in being able to track the process and to be able to understand what the other coworkers are trying to achieve and to check if the implementation was correct or not as described in section 8.1. The participant mentioned that most of the traceability they currently use and have comes for free based on their tool.

Of the interviewees that disagreed that the costs outweigh the expected benefits. One of the similarities is related to P1 and P2, in which even though the benefits may outweigh the costs, much of the realization of the benefits and traceability itself is also dependent on the understanding, decision, discipline, and intrinsic motivation of the engineers since traceability is not enforced.

In case of P1, even though it is in a heavily regulated and critical safety area,

"Yeah, exactly. Well in the end it is a quite, let's say, decision of every engineer, I do it or I don't do it. So, there is nothing imposed from externally that we have to do this but it is more let's say self-understanding that it makes sense, so nobody really complains about the costs. … So, it is the self-understanding that it makes perfect sense that what they are doing. "

In case of P2, the participant is of opinion that the benefits do outweigh the costs. However, the participant also mentioned that in his current environment the costs outweigh the benefits for what they do. However, much can be improved in that regard especially by reducing manual efforts. As described by P2:

"Yes. But, ... the benefits do outweigh the costs. But as a company we are not in the position to make these costs and it costs you a lot of effort to convince people to do so. And also, in the second, well I did a traceability study, so I know we can reduce the costs a lot and I think that is the way to go. We shouldn't force people to go the manual route basically. … I think you could say, at this point, the costs outweigh the benefits for what we do. But not for what we could do."

And when asked in regard to the need for a clearer overview of the traceability costs, P2 mentioned the dichotomy between the people who create the traces and the people who experience the benefits. As P2 stated:

"No. Yeah that is the second problem of course. The people who experience the benefits are not the people who make the costs most of the time. So, as a PO I am actually one of the only employees who experience both the costs and the benefits. Because I get a lot of benefits by searching and having everything related. But I am also responsible for making links in the development department so yeah. Support basically only have benefits if we do our jobs right and development, they only make costs. … There are some people who just, when they see a ticket, then they think 'oh this is related to something like that. … Jira has multiple types of links. So, you can have like blocking links, relation links, duplication, all kinds of stuff. And people make those kinds of links but most of the people don't. And it's also not a company policy to do so. It is just intrinsic motivation, again."

In case of P4, the answer was based on the same reason as given regarding the previous cost statement.

For P7, the participant mentioned that the costs were not an issue as similar to the previous cost statement and described that the benefits will help save a lot of time and effort as stated by P7:

"Yeah, I think, it is more guidance and keeping track of everything would definitely help. Even for a startup. Would help new developers joining the company so much. Just documentation or anything really. Keeping a record of what has been done in a project. Will definitely help any new developer joining the company. Because it saves a lot of time and effort really."

In case of P8, even though the costs are an inhibitor the participant basically mentioned in the previous statements and in section 8.1 the benefits of traceability for them and that end to end traceability is their main goal in which the benefits outweigh the costs. Especially considering keeping track of everything is very important for doing their job considering all the involved stakeholders, which holds true for many of the other participants as well.

Based on this we see that the answers on both the statements regarding the tradeoff of costs and benefits, and the need for a clearer overview are partially influenced in cases where the benefits are simply not clear while the costs are more apparent. Furthermore, the realization of the benefits also depends heavily on the view of the person responsible considering that in most of the cases of our participants traceability is not enforced. Furthermore, the different answers can also be attributed to that the people realizing traceability might not always be the ones experiencing the benefits.

Data collection and access

4.1	It is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability

4.2 There is a need to have easier access to information sources to be able to establish and maintain traceability

For the participants disagreeing with the difficulty in accessing and obtaining information sources, in several cases the reason behind the answers is simple. Some of the participants disagreed because they were the person responsible for traceability and had access to all the related artifacts. As described by P1:

"Well, I am the expert for this component. There isn't another person in the world that knows the system better than me. So, there is no need to access other people and the artifacts that I am dealing with, well, both the requirements and test cases are written by me so, or the signal list or the diagnostic services. So, as I write them, I have immediate access. So, there is no problem to get access to them."

And or in some other cases, they did express that it could be difficult in certain situations to get the right information from people, but if it was readily available and stored in the tools, it would be relatively easy to access it. As exemplified by P7:

"Yeah so in terms of the people regards yeah. Like as I mentioned, because the previous developers had left. It was definitely difficult to get a hold of them for getting information about what is happening in the project. There were some things that were left. For example, some other coworkers there, they would update me on the project. And certain other things were kept track on with certain things such as Trello board. "

On the contrary, reasons given as to why it would be difficult to access and obtain the information sources would be partially pinned down on environments where information was not centralized because of the use of multiple different tools and people. This could for example be in fast moving environments such as startups as described by P3:

"Yeah, sure. I feel also like, in general, information is all over the place. So that was also my experience in the other startup I worked at. Where, things come from different sources, like different people requesting stuff. And then there is like different information … it was also my experience in this other startup I worked. That information is like in different sources and different tools. So, you really need to make a conscious effort to get data from all those different sources and link that together if you want. "

Or in the case of larger environments where there is usage of a lot of different tools as well and with that also a lot of different user rights as to who have access to what information, as described in an example by P8:

"The problem is all the different artifacts are related to different user rights. I have just, two product owners in training that I am guiding through the process. And they started one month ago, and every day we are having the situation where they cannot do their work if they do not have the access rights and I first have to request the separate access rights so, it is separated and not everyone has access to everything."

Furthermore, similar as to the reason why it might be difficult in obtaining information of people as described by P7, P4 also mentioned that it is sometimes difficult to contact the right people externally and internally. As P4 described:

"Well, sometimes it is just difficult to find it back. Sometimes it is difficult to contact the right people internally, and of course we also have three different partners that also develop for us. So sometimes the knowledge is not even in-house but external. So, it is just difficult to get a clear overview of where that knowledge is within the organization. Within and outside the organization."

Based on this, we see that the need for easier data collection and access is influenced by the role and the tasks of the participants. In some cases, there is simply no need for easier access considering that most of the artifacts are already stored centrally and are easily accessed. However, the need for easier access and information gathering arises when there are multiple tools in play and in bigger environments where access rules play a bigger role or where the knowledge is external.

Managed

5.1	Traceability is mostly performed in an ad-hoc and non-managed fashion
5.2	There is a need to perform traceability in a more managed fashion

As to the reasons why participants agreed or disagreed that traceability is mostly performed ad-hoc this would partially depend on which artifacts are traced and the tooling itself. As described by P3 agreed:

"Yeah exactly. I think it depends whether we are talking about planned features or just requirements that just pop up like bugs, or things that we need to fix. So, I think that, if you are talking about features that were planned and kind of went through this process of refinement and estimation and everything. Then I think it is easier to trace back to where it came from. Whereas, if it's more like ad-hoc requirements, or things that popped up because we figured out that we needed to fix something. Then, zero traceability."

And as described by P2 who disagreed:

"What we do as for traceability that is very structured. I mean. Between business ticket there needs to be a product management ticket, then a development ticket, then a developer task and then a QA task. That is structured and that works. But providing additional context to a development ticket by relating it to other tickers and what is involved with that and all the bugs that people have reported, that doesn't work."

And as described by P6:

"Yeah, so the systems are in place. But when you do it, it is based on an ad hoc investigation. So, something went wrong in prod. Here is the thing, who has changed this code, okay I will go and ask them. So, it is kind of an ad-hoc situation but the tooling and everything is ready for you. … and the tooling in place is important for the different compliance aspects. For instance, so there has to be a managed process to get the tooling in place right, and there have been strong rules there. But when you're using it like after the fact. When you are actually looking at it then it is very ad hoc."

In addition, it would also depend on the ways of working that could differ per team as P5 stated:

"And so, it is based on the people. So, these people weren't used to writing down all that extra information and I used to. I have changed teams twice now, so I used to be on a team that involved a couple of those people. And they were very driving their feet of doing those things cause like, oh its busy work, this is like, you say this audit thing, but I have not seen it yet so. It is not really going to be

a big deal. And so yeah. I don't know. When I think of managed then you have either like this software that ensures it is being done or you have this hierarchy, this manager making sure that it is being done. And for most, of how I know of my current company does it per team. So pretty ad-hoc. "

Furthermore, a common theme was also simply that people were left on their own devices and had the responsibility and choice to do it or not to do it. As P7 stated:

"Yeah. The developers are left to do it on their own. "

And reasons for wanting a more managed process, even though it is not necessary in case of P7, it was to make it simpler for newer people that joined. As exemplified by P7:

"So, I put should have. I mainly didn't put must have because of that fact that it wasn't completely necessary. As we aren't a super big company where it is very necessary to have it in a more managed fashion. But I put should have in the fact that in my opinion. When I did first join the company, I did struggle to get good up with the project at first. Or it just took more time than I think it should have. Even if it just had to do with comments in the code for example or how things were documented. I think that should have been in a more managed fashion. Just to really make it simpler for newer developers joining or just yeah, new hires that want to get up good with the projects."

Furthermore, in some cases it depends on which environment and area the participant is responsible for and also related to changing environments and the way companies have grown. As described by P8:

"This is related to our changing environment. It was not managed as we started, and as we are right now in the changing process, we still have areas that are not managed. And we have other areas where we are changing, and we do it when we need it. … Yeah so ad-hoc. I have now a problem and now I need a solution, while the concepts were in place from the very first moments. The concepts were even in place before we started."

Furthermore, in the case of P1, most of the tools and managerial rules are in place already.

"… we evaluate traceability on a regular basis. Do we already have covered one hundred percent of my requirements or no we are still at the 85. Okay what's your plan, and until which point in time do you want to have fixed the gaps. When do you achieve one hundred percent traceability. So, with respect to that, that's used, yeah that's part of the regularly management discussions. What's the progress of the project so yes. And that was the reason I said strongly disagree. Because ad hoc means to me I just decide what to do in a certain situation and some weeks later I discard this decision and here it is pretty clear between which items there should be traceability and we have a regularly look on the, well, let's say traceability degree. Do we have one hundred percent coverage of my signals, of my test cases, of my requirements, yeah."

However, important to note is that even though traceability can be managed, in case of P1 it is also not necessarily enforced. As P1 exemplified:

"No, this is not enforced by top level management. But usually the single engineer asks, well especially if their let's say if there are external engineering offices involved. They use this as some kind of KPI's. If they write a test specification, okay now which coverage rate do I have meanwhile achieved? So, they have a very regular look on it. For the reports, there are companywide available reports. But you have to initiate this report on your own. So, there is no top level management intervention. If you decide in your project to skip traceability there wouldn't be at the first glance a

manager who blames you. Maybe, it might be if you have a highly safety critical system there will be a safety assessment at some point in time and there it would become obvious that you would miss traceability and then, of course, but if you have an let's say non safety relevant system which is also part of the vehicle and you wouldn't have traceability nobody would blame you. But usually, we maintain traceability nevertheless because we have the impression that it makes sense."

Based on this, the answers to the statements can vary based on the artifacts that are traced and the setup of the tooling itself. In addition, it can differ depending on the way of working per team and depending on how the responsibility of traceability is given, e.g. if people are responsible to do it on their own or if there are specific people that focus on it as their main task. Furthermore, it is also notable that even though traceability can be a more managed process, it might not be enforced in which people are still left with the choice to do it or not. And that it will depend on the motivation of the people to do it or not, which could for example be motivated in general by the area in which they work where safety and transparency can be important or just motivated by the benefits themselves.

Stakeholder views

6.1 Traceability is of high importance for the software development process
6.2 There is a need to increase the awareness of the importance of traceability

When asked to elaborate their answers on the importance of traceability for the software development process and the need for more awareness, most of the participants mentioned benefits that were also apparent in the literature and they also mentioned that traceability is important because it either is part of their job or because traceability can support them in their job. As described by the following participants:

P1: "*Yeah, it just helps you do your job. Otherwise you are getting lost in too much data. "*

P2: "*Yes, so the more context you have to a Jira ticket the more information the developer has, the better, the better the development usually goes. Because if I have a ticket to build feature 'a' and we see a relation to bug 'b', then they know, oh wait I need to take this customer scenario into account. Like what happens if the Wi-Fi drops out or anything like that. Just, yeah, it gives more context and that always helps. But yeah software development process would work without it of course but it would be not as good. "*

P3: "*Yeah, so. The benefits I see the most are really in those first phases of the process. If I am able to quickly see what customer feedback or like what problems link to different features. I think that is really valuable. Because than you can see if you are actually solving a problem or not. But then, yeah, I struggle a bit to tell you more about traceability in the later stages. "*

P5: "*I guess traceability in that regard is very useful. If you have a connection between the software that you wrote and the tests that you wrote. So that you can see, all of these things trace to each other and this software or whole feature set doesn't have any test at all. So, if you have that traceability then you see, oh there is no tracing here. Oh okay, then we just write test and the like."*

Furthermore, when asked about the awareness of the importance of traceability, most of the participants also mentioned that there are many mixed views on the importance of traceability. This could be due to managers just not being well versed in the more technical area or because people are not aware of the benefits or are just not simply aware of traceability at all.

P2: "Yeah, that ties into the same thing we discussed earlier. People who make costs do not experience benefits. If you do not experience benefits than why would you make costs. It requires another thought level to say, oh okay than my colleagues upstairs can than see, experience benefits of course. It is also not; it does not state who created the trace link right. It just, there is really no reward for people doing it."

P7: "Oh yeah definitely. Cause, especially since my boss isn't as well versed in software development or things like that. The opinions do differ. But he does take my opinion into account a lot when it comes to what I believe is best. For keeping records and things yeah. "

P8: "What I see from my perspective, is that most of the people don't even have a clue why traceability in general is needed. And if you don't have an idea about traceability, you don't have the awareness. And therefore, there is someone needed to tell you what traceability is in general, why is it needed. It can be part of the training, but I think it is also related to the general setup. And we have in our company, computer scientists, we have people coming from only mechanical parts, we have people from logistics, so not everyone heard about traceability before."

Based on this, we see that most of the answers logically depends on the goals and tasks of the participants and if traceability is part of their main job or if they perceive that it supports their job. Most of our participants see the importance of traceability, even though not all benefits might be clear. However, based on the interviews it became clear that there are many Mixed opinions in regard to traceability and that awareness of the benefits and or the concept as a whole plays a big part.

Allocation of resources

7.1	The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability
7.2	There is a need for more staff, time, and resources to properly establish and maintain traceability

As to the reasons for agreeing with that time, staff and resources are often insufficient, based on the interviews it was mainly contributed by not having enough staff, and that time and resources are highly related to it as well. As described by P8:

"It is a combination. As general, as company you first need a budget. If you have the budget you can hire people, or you can sign people to this project, or setup. So, then you have more staff. And if you have more staff, then you can have more time per person available to do the work. But work will always be the same so, if the budget is low, if the resources are low, what we need to build is still the same. So, they are highly related. "

As exemplified by P1:

"Yeah. Usually the staffing is well, the resources are well pretty small and of course, so there is always the risk that even if I have the understanding that this activity is absolutely necessary and I can benefit from it pretty immediately, I sometimes do not have the time to maintain traceability. "

And further exemplified by P2:

"Basically, the vast amount of Jira tickets in our company are created under time pressure because of things that have exploded basically. So that of course puts a time constraint on it and it's not like we can sit down and see ahh how can we improve this ticket with traceability. It is just, okay it is

good to go. …, we're very limited in product management department. So, when I joined, I was the second product manager/owner and now we are in the process of hiring our fourth product owner. But yeah, it is getting better."

In multiple cases the answers seem to also indicate that, even if there were enough time, staff, and resources to establish traceability, that it also depends on the roles and the responsibilities of the person, their discipline, and also their intrinsic motivation. As described by P3:

"Yeah that is definitely one thing. But I am also not sure that if we had more people, we would have more traceability. … I think it is a part of your product, or part of your role. It is not like there would be people focused on doing that. And I think that, depending on your role you are kind of responsible on how to do it as part of your job. But I think for many people it is just not a priority and they just end up not doing it. But I can imagine that as the company grows and there are like more teams. And you kind of need to communicate across teams. That those things would become even more critical. "

And exemplified by P7:

"And the allocation in the startup, as the developers are left to almost maintain their traceability themselves, so I felt like it uses up a lot of our time. We are already spending a lot of time essentially developing. You know, writing code for the project, doing all sorts of other development work and also having to maintain and manage traceability also. It definitely does take up more effort and time."

Furthermore, the answer on this statement also of course depends on the tool setups themselves as well.

Based on this we see that the time, resources, and staff are related to each other, in which the one affects the other. However, when there is a shortage it is usually directed to not having enough staff. Furthermore, even if there is enough staff or if more time, resources, and staff would be thrown to solve the lack thereof, it does not mean the problem the lack of traceability is fixed since it also depends on the people themselves and their motivations. In addition, having enough traceability and or no need for more time, resources and staff is also attributed to having the right tooling that provides traceability.

Collaboration

8.1	There is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability
8.2	There is a need for more collaboration between involved stakeholder teams regarding traceability

Regarding the answers on these statements, based on the interviews, the need for more collaboration, it depends on the structure and size of the company. In some companies, especially in smaller environments, informal verbal communication is enough. As described by P3:

"And also, because it is such a small company. In those cases, verbal collaboration kind of replaces this traceability. But yeah, probably in bigger companies I think it is a bit less informal. "

However, this is of course in trade off against more formal collaboration between people and teams. As described by P3:

"There is already very constant communication and collaboration. And because these happens, and a lot informally, I think we end up having not that much formal traceability. Because we kind of like working together. And we know things because we are close."

However, this can become of bigger concern in larger environments where formal collaboration between teams become of more importance such as in the case of P8. However, in some cases, it is not necessarily a problem depending on the setup of the company structure itself and with Clearly defined roles and responsibilities. As described by P2:

"basically, what I meant here is to say, the collaboration between for example the development department or the support department or the consultancy department, there is no collaboration there. This is also by design by the way. We try to keep the development department really closed to focus on what they need to do. The product management department basically also works as a gatekeeper there."

In addition, the need for more collaboration can stem from trying to align the involved stakeholders and teams. As described by the following participants:

P1: *"… if multiple persons are working on traceability, there is always, let's say, some rest risk that is somehow different understanding on how to use traceability on a deeper level. So, of course, there is, it would make sense to collaborate even more than they do. But it is nothing that I would treat as problematic."*

P4: "*Yeah. At our place you mainly notice that the user has a distance. And because of course you have a manager and there is some stuff between and then eventually the It department. You also notice that the process is followed very closely, that basically results in that you are very stuck or focused on the process. And ideally you would, yourself with the user sit together. Especially if you are managing or create a product backlog. Then you can be like, oh I talked to the PO of mortgage etc. But see the PO as non-technical. E.g. if papers have to be changed and so on. But you also want a PO in the area of IT. That is kind of missing here.* "

P7: *"There is some, somewhat. We do, as with really having meetings with stakeholders, so there is somewhat, some form of communication in relation between the teams to establish some form of traceability and some things of how should we, what should we do to record this data and what should we, what things, what apps should we use. So, there is yeah somewhat. … it is mainly to keep everyone on the same page on what we're using and etc.* "

Based on this, the answers can vary based on the size and structure of the companies. In smaller companies there is less need for formal collaboration and communication in regards to Traceability compared to larger and more structured companies. Furthermore, in cases where there are practices such as team meetings or any form of collaboration it is also mainly to align everyone on what they are doing.

Responsibility

9.1	It is unclear which roles are responsible for traceability
9.2	There is a need for more communication about who are responsible for traceability

Regarding which roles are responsible, in most cases of our participants, the responsibilities of each role is clearly defined in the companies and the responsibilities of traceability. In some cases, however, it is more unclear. This would come up in environments where for example where the roles are more flexible and therefore also the responsibilities. As described by P7:

"I think the biggest unclear part as, because the roles are very flexible. My role is currently unity developer. But I have done a bunch of other things. Of the top of my head I can't think of anything but yeah. A lot of things outside of unity for example. I guess that is the issue. It is very unclear on what our roles are. So, it is very flexible. We can, everyone has done a bit of everything almost. "

Furthermore, the responsibilities can become unclear as well in fast growing environments where new roles and employees are introduced which could lead to overlap and or lack of defining where they are responsible for. As exemplified by P8:

"We started two years ago with five people. We are now working with something between 150 and 200 people. So, within two years we went from 5 to more than 150. This leads to the situation that the first setup, several people were doing a lot of different aspects. And then new people were introduced without defining what is their area of responsibility. So, we now have overlaps that we need to identify, and we have gaps, things that are not covered anymore. So, I think it is related to the fast growing and it is related to the changes."

Furthermore, P8 also mentioned that another factor could be because of cultural differences. As described by P8:

"Maybe it is also related to our company, what I see is there is a cultural difference in general. Young people tend to plan everything, every single detail up front and are wondering when something is not working according to plan. What I see in the Netherlands. Netherland people, yeah, they are planning to a specific amount of degree. But at the end they are just doing. They are just solving problems and they are just fixing issues. So, if the German people in our setup are planning a lot. They are defining what is a role. The Dutch people will already see that there is a gap and they will do it. And then we have an overlap. … So, there is also an intercultural aspect and within here it is the same. In India you need to define what someone is allowed to do. If you don't define it, nobody will do it. "

Based on this, we see that in cases where the responsibilities of which roles perform traceability are not clear, is in cases where the responsibilities of the roles themselves are also not clear. This could be due to being in a flexible environment such as startups where everyone does a bit of everything, or it could be in the cases where the responsibilities of multiple overlap each other. In addition, this seems to be also linked to cultural aspects, where in some cultures there is more of a do vs a more planning mentality.

Tools

10.1	Traceability tools do not satisfy our needs
10.2	There is a need for better traceability tools

As to the reasons given of why participants agreed with the statement that tools do not satisfy their needs.

In case of P1, the main tools they use or have their main traceability in is Doors. Even though Doors has some links to other tools such as MATLAB, most of the traceability artifacts end up in Doors. The main issue P1 has with the tool is that it is difficult to handle artifacts and to get a more complete overview of the traceability picture. As described by P1:

"Yeah, it is mainly the handling stuff. If you ever have to use Doors, working with traceability it works, it is okay. But of course, it could be better. If you think about drag and drop activities, on visualization and so on. Well, usually it is not that easy to see at one glance what is the overall traceability picture. … The individual traces are there, and you can jump forth and backwards. That is okay. But if you for instance would like to annotate something to this trace. Well it works but, and especially the overview, so it is nothing that is let's say, treated or considered as fun. "

In case of P3, they currently use Trello for everything related to task management and have extensions to integrate Trello with GitHub. The main complaint here is that when performing traceability tasks such as linking artifacts together e.g. epics and user stories and features, there is a lot of manual work involved. Especially in the case when a variety of other artifacts such as design documents and so on are stored in a variety of other tools. As P3 described:

"Yeah so one of the things I struggle with, with Trello, for linking stuff together. Is that, for instance, like. You typically have epics and user stories. Where one epic might be broken down in different user stories. … And then if you want to do that in Trello. It is like all those different things they are different cards. So, I have one card for the epic, and one card for the user story, and then each user story has a number of tasks to implement the user stories. And tasks are also different cards. So if you want to link them all together, what you need to do, is basically, literally like copy paste the Trello card URL, and go card by card, linking that."

In the case of P8, it is mainly because none of the tools used is built to be used in the entire chain setup of tools they currently use. As for the reason they use different tools, this is mainly because each of the tools offer different levels of detail than the other tools. As described by P8:

"And we are using different tools as we have more detailed need on each level than one of the tools is offering. We have HP quality center in place, which is in general an application life cycle management tool that should have the possibility to trace over the complete process. But, in requirements tracing we are missing aspects that are related on just multiple documents level, multi split of modules of software components of software and hardware components, material handling equipment. So, we are using tools for a specific area and then the combination of these tools is not supported. "

In which the participant said that it basically comes down to not having a proper interface between all their used tools to be able to share information between these tools without too much effort.

Regarding the participants that disagreed, the answers were mostly straightforward since most of the tools do what the participants need to use them for. However, even though the tools satisfy their main needs, most of the improvements are also directed to minimizing the effort and increasing automation around these tools. As described by P2:

"I mean, the Atlassian suite does what it needs to do right. It allows you to create every link you could possibly dream of. The only thing is that you still need to do it manually. And I don't think they consider that their responsibility. So that's why. Jira, confluence, and other tooling in that suite that is pretty good. It is just getting the trace links in."

Based on this, in most cases of our participants, the tools that are used satisfy their main needs and tasks. However, most of the needs seem more related to the quality aspects in which there is an emphasis on reducing effort as to inputting trace links and linking different artifacts and or different tools to each other, which could provide a better and overall traceability picture.

Storage and versioning

11.1	It is difficult to establish traceability because development artifacts are stored in multiple locations/repositories
11.2	There is a need for a more centralized development artifact repository to establish traceability more easily

Reasons for disagreeing is in most cases also straightforward in which most of the artifacts are stored centrally in tools or where they make use of tool suites that easily connect to each other, as described by the following participants:

P1: "*No, they are stored in Doors. So that is the reason why I indicated somewhat disagree at statement 11. The majority is in Doors, let's say, except the MATLAB models. So that is the reason for somewhat disagree. There are of course some external resources, or you would like to refer to supplementary specification document which is stored in an external document so there are some things that could be better. But, as it is very Doors centric right now, it is not a big issue.*"

P2: "*Yeah exactly. If you use the entire Atlassian suite, so Jira, bitbucket, confluence, then you have no problems there. You can link from all the way from business ticket to development ticket and then you can find some documentation in confluence and you can see how it was tested in test rail which is not from Atlassian but it's a different tool. But you have a Jira plug in for it so that also works. Yeah, that part works pretty well. No problems.* "

In cases where it was difficult to establish traceability it was mainly because of usage of a variety of tools and lack of interfaces as described in the previous topic.

Complexity and integration

12.1	Tools used for traceability are too complex to effectively use and to integrate with our existing tools
12.2	There is a need for less complex traceability tools and easier integration with our existing tools

Most of the participants indicated that most tools are not too complex to use and to integrate with their existing tools. Reasons given was that in several cases, tools of a similar suite were used and therefore of course there are no problems with integrating tools.

In addition, in some cases traceability was just a byproduct of the main tasks they were performing in the tools which they were familiar with in which there is no need for less complex tools, as was described by the following participants:

P1: *"yeah. It is the tool for the requirements, so with or without traceability this is the tool to use for writing specifications. So, there is no add on. It is not like an external traceability tool that I use to plug in to existing engineering tools. The tool and one of its functionalities are providing traceability so it's not too complex because the tool is used anyway. And there is no add-on to something because all the traces are handled in this tool. "*

Furthermore, the complexity also simply depends on how familiar people are with the tools and the awareness of other potential tools, as described by the following participants:

P6: "No*, they are pretty advanced. It is not known for being easy to use. But it is standardized open source tool which people use in many different companies. So, you start to learn it but it is not an easy to use tool. It is an advanced tool. "*

P8:" *Yeah. So, uhm. They are too complex to use, is I somehow agreed what I know of these tools, I see that is a problem, but I am totally aware that there are much more tools that I don't know or that I don't know in such detail. Maybe there are tools on the market that would fulfil this, I just don't know. "*

Finally, it could also be due to lack of awareness of the features in their main tools and the use of multiple tools which added to the complexity. In case of P8, the participant mentioned that not everyone in their work environment is familiar with all the features in their tools.

Automation

13.1 Traceability is mostly performed manually

13.2 There is a need for more traceability automation

In most cases, traceability is performed manually and there is a need for more traceability automation to reduce repetitive tasks that require a lot of effort such as in the case of trace creation and or maintenance, as exemplified by the following participants.

P3:" *Yeah so one of the things I struggle with, with Trello, for linking stuff together. Is that, for instance, like. You typically have epics and user stories. Where one epic might be broken down in different user stories. So you have one epic that is, I want to find hotels, and then this breaks down into different user stories like, okay I am going to search for an hotel and filter hotels, I am going to do I don't know what with the hotels. And then, if you want to do that in Trello. It is like all those different things they are different cards. So, I have one card for the epic, and one card for the user story, and then each user story has a number of tasks to implement the user stories. And tasks are also different cards. So, if you want to link them all together, what you need to do, is basically, literally like copy paste the Trello card URL, and go card by card, linking that."*

P5:" *yeah. And I guess, we sort of have, it is not automated yet but at least it is 'less typing'. Is what we had in our Sprinter, and in our -Company E- platform where you write a commit and a pop up comes up if what you are working on is connected to the project correctly. It will show you the list of the names of the stories that you have worked on and then you have checkboxes. So instead of going to your jira board, seeing what the code is, then going back to your commit message, and typing in the number, and hoping that it is the same, because you might have a typo. You just see the list and*

it sees the title, so it says add button to page, and then a checkbox. And then you can just check the checkbox. So it is not completely automated but it reduces it to one click. So, it makes it easier. "

Even though most of the participants would like to see things more automated. Some of them also mentioned that a fully automated system would be difficult and that there should always be a person involved to check the correctness. As described by the following participants:

P1: *"Right now all the traces are maintained manually. And some years ago, I had a PHD student who was working on deriving trace links between requirements and test cases automatically. So, we did some work with Jane, Cleland, and so on. We tried to figure out if there is a chance that we are not forced to maintain the links manually. But can derive them automatically. And although, he made significant progress. In the end, the precision and recall values were at a level, well, that I would say, it doesn't make sense to use them because of how many false positives and nobody could rely on this data."*

P3*: "Yeah, I think it is always tricky. I doubt you would have like a fully automated thing. But what I think about when I think of like automated traceability. I am thinking more around like, again like, doing this process of things coming, coming up from problems till features and production. That it would be just easier to, like, it is almost like you start with this big thing and as you break down stuff. If they were somehow linked right away. For instance, you start from this big problem and then you come up with some epics, and that I don't need to enter manually that his epic solves this problem. And then the epic will break down in user stories and then those user stories are like linked to this epic. So, I know that, in Jira, for instance, it is a bit easier to link epics to user stories more or less. Well it is not that easy to make; you still need to link them manually. But it is easier to see like what user stories trace back to which epics. Whereas in Trello, it doesn't exist. Like, you can link them, but if you want to see like okay, for epic, what are the user stories that are linked to them. You cannot do it. You can only see like the individual links basically. "*

P7:" *I think in my opinion. Yeah automation is definitely nice for the small things. Like attaching the ID, putting it on the board etc. But I personally believe having a human eye to look at it as well to maybe iron out certain things that the automation may have missed. Yeah it is always good to have a human eye to look at it instead of having it fully automated.*"

Based on this, reasons for performing traceability manually and having a need for more automation could be because it is part of the tools and the way of working of these tools. Other reasons also because of the low recall and precisions of automated tools that have been tested and are not yet on a level to be used in an actual setting.

9. Results and Discussion

In this section we describe and discuss the results to the sub research questions based on the literature review and questionnaire, relate the findings of the questionnaire to interviews and the literature, and discuss the limitation opposed on this study.

Recall as described in section 1.2, the goal of this study is to investigate the perceived value and needs of practitioners in practice and differences between the perceived value and needs between Agile and non-Agile SDLC, and critical and non-critical projects.

To help achieve this objective, we formulated the initial research and sub research questions, as described in section 1.4, to understand the context and theoretical background of software

traceability, which was answered by executing a mixed snowball and database search literature review.

Based on the results of these questions the main research objective and research question was refined and further sub questions were specified as described in section 1.4.

These research questions were to be answered by executing the empirical part which is mainly by means of a questionnaire. In addition to this, to gain more insight and to understand more of the rationale, we performed interviews with the participants.

9.1 Results

In this section, we describe the results for the research questions as stated in section 1.4.

RQ1	"*What is the current state of software traceability in research and practice*?"
RQ1.1	"*What is the current state of research regarding software traceability?*"
RQ1.2	"*What is the current state in practice regarding software traceability?*"

Based on the literature review, it became clear that there are still many issues and challenges regarding software traceability in practice, in which many of these issues and challenges were identified by empirical research conducted several decades ago and which are still being conducted as of this day. At this point, much of the traceability research is focused on tackling these issues and challenges which spawned initiatives like CoEST and many other researches that work on the identified challenges as stated in the Grand challenges of traceability. What is noticeable is that much of the current research is focused on reducing the costs, in money, time, and effort, regarding applying traceability in practice. This includes research on traceability artifacts, methods, and tools to more reliably, accurately, and more effortlessly create and maintain trace links between a variety of artifacts and in artifacts themselves for use by practitioners.

Even with all these efforts and all the progress made so far, much still must be done. In addition, we still noticed that there is a gap or imbalance between research and practice, in which the research community indicated to be still in need of more contemporary and empirical evidence and data regarding traceability in practice, especially outside of the typical critical safety projects and Traditional development environments.

RQ2	"*What is the reported value of software traceability in the literature?"*
RQ2.1	"*What are the reported benefits of software traceability in the literature?*"
RQ2.2	"*What are the reported costs of software traceability in the literature?"*

According to the current literature there are many benefits and much of the value and benefits of software traceability lies in the traceability related tasks that traceability can enable, such as impact analysis or being able to trace the origin and rationale behind artifacts and the artifacts themselves, documenting a requirements history, and tracking requirement or task implementation state.

In addition to the tasks that it can enable, traceability can also improve the development processes by providing process related benefits such as increased time to delivery, increased transparency, and increased reliability, however these are much more difficult to prove.

Of course, traceability does not only come with benefits. According to the current literature it also comes with a lot of costs in time, effort, and money. Several studies have found that practitioners have difficulties in expressing the benefits while the costs are more apparent. In addition, the

perception on what traceability can bring to the table is heavily influenced by all the issues and challenges found in prior studies, in which for example the cost-benefit fallacy partially influences the views of practitioners since the person establishing and maintaining traceability does not always get the benefits. In addition, the ad hoc nature of performing traceability in practice as found by researchers also make it more difficult to realize the potential benefits of traceability and most of the current tool support do not yet have reliable ways to reliable and accurately establish the necessary links.

As such, even though there are many benefits to be expected from software traceability, actual evidence is scarce while the evidence of costs is clearer and more apparent, which intensifies the varying effort put in managing a traceability process in practice.

RQ3	*"How can the perceived value of practitioners regarding software traceability be measured?"*

According to the literature review, perceived value is an extensive researched concept in the marketing and consumer theory and there are multiple research streams and different ways of viewing and measuring the concept. Typically, perceived value is viewed from two common perspectives, which are the unidimensional perspective and the multi-dimensional perspective.

One of the simpler ways to look at perceived value is in terms of its perceived benefits and perceived costs in terms of price and or functionality. This kind of view is usually adopted in price-based studies where the worth of the product or service is translated to a certain price tag. In addition, it is also adopted in means-end theory studies where the worth is based on how a certain product or service can help in achieving a goal, means, or end that the consumer has in mind.

In contrast to the uni-dimensional view, the other view perceives the construct of perceived value as a construct that is made up of other and more complex underlying constructs. It can for example be based on a 'value hierarchy model' in which the perceived value can be measured based on consumption goals, consequences, attributes, and desired value and received value. On an abstract level, this view will contain more concepts and constructs that make up perceived value than just for example the benefits and costs. The strength of this perspective is that it adds more richness to how perceived value can be viewed. However, because of the introduction of extra concepts and constructs, it also becomes more complex.

Therefore, there is no one size fits all way to measure perceived value and different views come with different pros and cons.

As such, the view that we have chosen to adopt during this study regarding perceived value is more closely related to the means-end theory perspective and the definition of perceived value itself. This means that when perceived value is mentioned in this study, it refers to the construct of perceived value which is made up based on the needs. However, it is important to understand and be aware that this is not the only possible way to view and define perceived value.

In addition to measuring the perceived value based on what is needed of traceability and their ranking based on priority and importance, this study also measures the context e.g. current situation of the participants, since it is important to understand the different contexts and how they affect the needs of practitioners. In other words, in this study, we will be measuring perceived value in terms of the current situation of practitioners and in terms of their needs.

RQ4	*"Is the perceived value of software traceability affected by the type of software development life cycle?"*
RQ4.1	*"Is the current software traceability situation affected by the type of software development life cycle?"*
RQ4.2	*"Are the software traceability needs affected by the type of software development life cycle?"*

Based on the answers of the sub research questions, the results indicate that the SDLC significantly affects the perceived value of software traceability regarding their current situation of performing traceability manually. Surprisingly, we expected the SDLC to be significantly affecting more of the perceived value, but instead the SDLC seemingly does not affect the perceived value of software traceability significantly in most cases. This implies that overall between the three groups of SDLC that their perceived value regarding their current traceability situation and their needs are similar to a certain extend.

The reason why it might be similar to an extent is because, even though there were almost no significant differences found according to the statistical analysis, we did see differences when looking at the sentiment of the participants of the three groups.

When looking at the sentiment regarding their current situation we see that the participants of the three groups had similar responses on the topics of costs, importance, resources, responsibility, and automation. Most of the participants of all three groups agreed on that costs were the main inhibitor, that traceability is of high importance for the software development process, that the allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability, that it is unclear which roles are responsible for traceability, and that traceability is mostly performed manually. In addition, most of the participants of the three groups do not think that the traceability costs outweigh the expected benefits, and that the tools used for traceability are not too complex to use effectively and to integrate with their existing tools.

The differences in their sentiment was related to the topics of guidance, data collection, managed, collaboration, tools, and storage and versioning.

The difference regarding most of the participants in the Agile group, in contrast to the other groups, is that they did not find it difficult to access and obtain the information sources to establish traceability, and they indicated that their traceability tools satisfy their traceability needs.

In relation to most of the participants in the mixed group, in comparison to the other groups, they indicated that there is no lack of collaboration between involved stakeholder teams regarding establishing and maintaining traceability.

Finally, for most of the participants in the Traditional group compared to the other groups, they indicated that there was enough guidance, that they do not mostly perform traceability in an ad-hoc and non-managed fashion, and that it is difficult to establish traceability because development artifacts are stored in multiple locations and or repositories.

Furthermore, when looking at the results regarding their sentiments and on which statements each group agreed the most on it becomes clear that all three groups have similar sentiments. Most of the participants in the Agile group agreed that traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices, followed by that traceability is mostly performed manually, and that most of the participants in the Agile group indicate that

traceability is of high importance for the software development process. Regarding the mixed SDLCs, most of the participants in that group agreed that traceability is of high importance for the software development process, followed by that traceability costs in money, time, and effort are the main inhibitor, and that the allocation of time, staff, and resources are often insufficient to be able to properly establish and maintain traceability. Furthermore, most of the participants in the Traditional group agreed that traceability is of high importance for the software development process, followed by that traceability is mostly performed manually, and that traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices.

Now that it is clearer what the overall current situation is between the groups, we will focus on the needs of each group.

When looking at the sentiment regarding the needs of the three groups, most of the participants of the three groups had similar responses on the topics of costs, ROI, managed, different stakeholder viewpoints, collaboration, responsibility, and automation. Most of the participants of the three groups indicated that there is a need to have a more clear overview of the costs and benefits regarding traceability, that there is a need to perform traceability in a more managed fashion, that there is a need to increase the awareness of the importance of traceability, that there is a need for more collaboration between involved stakeholder teams regarding traceability, that there is a need for more communication about who are responsible for traceability, and that there is a need for more traceability automation. In addition, all three groups indicated that there was either no need or saw it as more of a low priority to reduce the costs of traceability.

The differences in their sentiment was related to the topics of guidance, data collection, allocation of resources, tools, storage and versioning, and complexity.

The difference regarding most of the participants in the Agile group, in contrast to the other groups, is that they did indicated that there is a need to have easier access to information sources to be able to establish and maintain traceability, and that the needs for more guidance, and the need for more staff, time, and resources are of less need or no need.

In relation to most of the participants in the mixed group, in comparison to the other groups, they were evenly split between the need for more staff, time, and resources, and most of the participants indicated that there was no or low need for better traceability tools and the need for less complex traceability tools and easier integration with their existing tools.

Finally, for most of the participants in the Traditional group compared to the other groups, they indicated that there is a need for more staff, time, and resources to properly establish traceability more easily, and that there is a need for a more centralized development artifact repository to establish traceability more easily.

Additionally, when looking at the needs between the groups, we see that the Agile group values and prioritizes the need to perform traceability in a more managed fashion as most valuable and needed, which is followed by the need for a more clear overview of the costs and benefits regarding traceability, and the need for more traceability automation. The mixed group values and prioritizes the need for a more clear overview of the costs and benefits regarding traceability the most, followed by a need to increase the awareness of the importance of traceability, and the need for more guidance regarding traceability. The participants in the Traditional group value and prioritize the need to perform traceability in a more managed fashion the most, followed by the need to increase the awareness of the importance of traceability, and the need for more collaboration between involved stakeholder teams regarding traceability.

What is surprising is that when comparing the sentiments regarding their needs and their current situation is that a part of the respondents in the Agile group indicate that there is a need to have easier access to information sources to be able to establish traceability, while they also indicated to disagree with that it is difficult to access and obtain information sources to be able to establish traceability. Additionally, part of the respondents in the Agile group also indicated that the need for more guidance and staff, time, and resources are of less need or no need while they indicated that there is insufficient guidance within the company on how to establish traceability and that the allocation of time, staff, and resources are often insufficient to be able to properly establish and maintain traceability

These results might seem contradicting however when looking at the questionnaire and how the statements and the interviews it becomes clear what the reasons are behind these seemingly contradictions, and also any contradictions when comparing safety-critical and non-safety critical, which are further discussed in section 9.2.

RQ5	*"Is the perceived value of software traceability affected by the type of software project?"*
RQ5.1	*"Is the current software traceability situation affected by the type of software project?"*
RQ5.2	*"Are the software traceability needs affected by the type of software project?"*

Regarding the Likert response distributions, based on the Wilcoxon analysis of section 7.2.1, we can conclude that there are no significant differences regarding the Likert response distribution of the perceived current situation of traceability between safety critical and non-safety critical projects.

Even though there are no significant differences found, similar to the topic of the different SDLCs, there are also differences regarding the sentiments related to their current situation of traceability.

Based on the sentiment comparison between the groups in section 7.1.1, the results show that most of the participants of both safety critical and non-safety critical project groups agreed that traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices, that there is insufficient guidance within the company on how to establish traceability, traceability is of high importance for the software development process, that the allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability, that the tools used for traceability are not too complex to use effectively and to integrate with their existing tools, and that traceability is mostly performed manually. Furthermore, both groups do not think that traceability costs outweigh the expected benefits, and that it is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability

The results do show differences regarding the sentiment between the groups. Compared to most of the participants in the non-safety critical group, most of the participants in the safety critical group indicated that traceability is mostly not performed in an ad-hoc and non-managed fashion, they indicated that there is no lack of collaboration, that traceability tools do not satisfy their traceability needs, and that they do not think it is difficult to establish traceability because the development artifacts are stored in multiple locations and or repositories. In addition, the participants of the safety critical group were split about that it is unclear which roles are responsible for traceability.

Furthermore, when looking at the results regarding their sentiments and on which statements each group agreed the most on it becomes clear that both groups have similar sentiments. Most of the participants in the safety critical group agreed that traceability is of high importance for the software development process, followed by that traceability costs in money, time, and effort are the main

inhibitor for adopting (more mature) traceability practices, and that the allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability. For the non-safety critical group, most of the participants of this group agreed that traceability is of high importance for the software development process, followed by that traceability is mostly performed manually, and that traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices

Now that it is clearer what the overall current situation is between the two groups, we will take a look at the needs of each group.

Regarding the sentiment between the groups there are differences regarding the perceived value of software traceability needs. Based on the sentiment comparison between the groups in section 7.1.2, the results show no difference between the sentiment of the groups for most of the needs. Both safety critical and non-safety critical indicated that there is a need for more guidance in the company regarding traceability, that there is a need for a more clear overview of the costs and benefits regarding traceability, that there is a need to have easier access to information sources to be able to establish and maintain traceability, that there is a need to perform traceability in a more managed fashion, that there is a need to increase the awareness of the importance of traceability, that there is a need for more collaboration between involved stakeholder teams regarding traceability, that there is a need for more communication about who are responsible for traceability, and that there is a need for more traceability automation. Furthermore, both groups indicated that the need to reduce the costs of traceability, and the need for a more centralized development artifact repository to establish traceability more easily, are of low priority or no need.

In addition, the results do show differences regarding the sentiment between the groups. Compared to the safety critical group, the non-safety critical group, agreed that there is a need for less complex traceability tools and easier integration with their existing tools, and they indicated that there is no need or a low priority need for more staff, time, and resources to properly establish and maintain traceability, and that there is no need for better traceability tools.

Furthermore, when looking at the needs between the groups, we see that the safety critical group values and prioritizes the need to perform traceability in a more managed fashion as most valuable and needed, which is followed by the need for a more clear overview of the costs and benefits regarding traceability, and the need to increase the awareness of the importance of traceability. The non-safety critical group values and prioritizes the need for a more clear overview of the costs and benefits regarding traceability the most, followed by a need to increase the awareness of the importance of traceability, and the need for more traceability automation.

Now that we have answered the sub research questions, it is possible to answer the main research question of this study:

MRQ	*"How is the perceived value of software traceability affected between Agile and Traditional development environments and between safety-critical and non-safety critical projects?".*

The perceived value regarding the current traceability situation and their needs of the participants and the effect of the SDLC on it seems to only be significantly affected regarding the topic of automation, if traceability is performed manually or not. In addition, the type of project also does not significantly affect the perceived value of traceability. Even though there are no significant differences, in this study we did identify differences and similarities in the sentiments between Agile, mixed, and Traditional environments as well between safety-critical and non-safety critical projects and also identified which needs are most valued per group.

In addition, even though the SDLC and project type regarding safety and non-safety do not significantly affect the perceived value, it is still possible that they have an affect but that there are other factors affecting the perceived value in a more direct way, which is partially discussed in section 9.2

9.2 Discussion & relation to literature

For this section, considering that the findings of the first few research questions are already derived from literature, we will focus on how the general results and the results to research questions based on the questionnaire are related to the interviews and the literature.

Contradictions

When talking about contradictions, in our case we mean statements where participants answered that there is a problem, but that they seemingly do not have a need or low need to improve it, or where there is no problem but where they still would want to improve it. Generally speaking, most of the seemingly contradictions can be attributed to the fact that the current situation statements and their related needs were looked at in isolation. These seemingly contradictions, can partially be explained considering that e.g. the need for more traceability automation covers and potentially affects multiple statements of their traceability situations and perceptions. Furthermore, even though there might be a problem, the need for something else might be higher considered we asked the participants to prioritize the needs based on the predefined categories as described in section 4. In addition, it could also simply be the case that several participants do not really perform traceability as seen in section 5.1.8, in which even though participants could perceive something as a problem or not, that there is just no perceived need to improve it since they do not really perform traceability. Next to these more general potential reasons, other potential reasons are also discussed in the following sections.

Costs

Looking at the results of all the participants, we see that most of the participants agree with that traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices.

When comparing the different SDLCs groups in relation to their current traceability situation and their needs regarding the costs of traceability, we see that most the participants of all three groups agreed that the traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices.

In addition, we also see that when the participants are grouped in terms of project type, that over many of the participants in both safety and non-safety critical groups also agree that the costs are the main inhibitor.

Surprisingly, over half of all the participants, and of the participants of these groups, also indicated that the need to reduce the costs of traceability is of low priority or no need. This, however, becomes less surprising when we take a look at the rationale of the participants that were interviewed.

According to the interviews, even though the costs are seen as an issue, the answer is not just reducing these directly costs but rather improving for example their current tools or their tool setup which eventually lead to reduced costs.

In addition, according to most of our interviewees, the main inhibitor is mainly the extra effort people must go through to adopt and maintain traceability, specifically in manually creating traces and maintaining them.

The results regarding the topic of costs indicate to support the previous findings in the literature. Similar to the findings and topics discussed as found in the studies of Egyed, Grunbacher, Heindl, & Biffl, 2007; Orlena Gotel et al., 2012 ;and Regan et al., 2012, most of the participants of our survey also indicated that costs are a main inhibitor for adopting (more mature) traceability practices and that traceability itself requires a lot of manual effort which in turn costs time and money.

Guidance

According to the results of all the participants, we see that most of the participants agree with the statement that there is insufficient guidance within the company on how to establish traceability and that there is a need for more guidance.

When comparing the three SLDC groups, based on our data more than half of the participants in both the Agile and Mixed SDLC groups also agreed to a certain extend that there is insufficient guidance, while over half of the Traditional group indicated that there is enough guidance.

Interestingly enough, when comparing those situations to their needs we see that over half of the participants in the Agile SDLC do not prioritize the need for more guidance, while the mixed and Traditional groups have a higher need for more guidance in the company regarding traceability.

Furthermore, both safety and non-safety agree that there is insufficient guidance, and both also have a need for more guidance.

Based on the interviews, we can see that guidance can come in varies forms. This could for example be in terms of trainings and lectures provided by the companies or simply in the form of knowledge passing from 'trainer to trainee' so to speak. Furthermore, even though there might be general knowledge passing, which seems to be also be generally lacking in regards to general tool guidance according to several of our interviews, the interview results also seem to indicate that traceability itself is not always included in the knowledge passing in regards to establishing traceability and how to use their tools for it, even if the features might be present.

The results regarding the topic of guidance and training seem in line with the findings of previous studies. Similar to the findings from Cleland-Huang et al., (2012); Mäder et al., (2009); Ramesh, (1998), and Regan et al., (2012), most of the participants indicated that there is a lack of insufficient guidance regarding traceability itself and on how to use their tools to establish and maintain traceability, which could be via training provided by the companies or via general knowledge passing from person to person.

Return on investment

Regarding the topic of ROI, most of the participants indicated that currently, the costs outweigh the expected benefits and that over half of the participants also indicated that there is a need to have a more clear overview of the costs and benefits regarding traceability.

When comparing the three SDLC groups, over half of the participants in each group indicated that traceability cost outweighs the expected benefits and that there is a need to have a more clear overview of the costs and benefits regarding traceability.

Furthermore, over half of the participants from both safety critical and non-safety critical projects indicated that the traceability costs outweigh the expected benefits and that there is a need to have a more clear overview of the costs and benefits regarding traceability.

According to the interviews, in some cases the benefits were not clear while the costs were more apparent. In addition, it was also mentioned that the realization of the benefits depends on the view of the person responsible since in some cases traceability was not enforced and relied on the discipline of the person responsible. Finally, it was also mentioned that in part it is also attributed to that the person realizing traceability might not always be the ones experiencing the benefits.

The results regarding the topic of return on investment indicate to be similar to the findings as found in the studies of Arkley & Riddle, (2005); Bouillon et al., (2013); Ramesh, (1998), and Wohlrab et al., (2018). Similar perceived problems were found in this study as well such as the lack of direct or tangible benefits and that the person realizing traceability does not always experiences the benefits, while the costs were more apparent. In addition, similar to the previous findings, much of benefits depends on the tool use but in addition also on the discipline of the people responsible considering that traceability is not always enforced, even in more managed and safety critical projects.

Data collection

In relation to the topic of data collection, over half of the participants indicated that it is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability and over half of the participants also indicated that there is a need to have easier access to information sources to be able to establish and maintain traceability.

When comparing the three SDLC groups, over half of the participants in the Agile group indicated that it is not difficult to access and obtain information sources, while mixed and Traditional groups indicated that it is difficult. Furthermore, most of the participants in the Agile group indicated that the need to have easier access to information sources to be able to establish and maintain traceability is of high importance while this was not the case in the mixed and Traditional groups.

Furthermore, most participants of both project types indicated that it is difficult to access and obtain information sources to be able to establish traceability and most of the participants of both groups also indicated that there is a need to have easier access to information sources to be able to establish and maintain traceability.

Based on the interview results we saw that the need for easier data collection and access is influenced by the role and the tasks of the participants. In some cases, there is simply no need for easier access considering that most of the artifacts are already stored centrally and are easily accessed. However, the need for easier access and information gathering arises when there are multiple tools in play and in bigger environments where access rules play a bigger role or where the knowledge is external.

The findings regarding data collection can be seen from two perspectives, collecting data via people or collecting and accessing data via artifacts.

Similar to the findings and topics discussed in studies from Gotel & Finkelstein, (1994); Ramesh, (1998), and Regan et al., (2012), we perceive the difficulty in accessing and obtain the right information sources regarding artifacts. This is in part influenced by the topic which is discussed in the topic of storage and versioning. However, the difficulty increases when multiple tools are in play and in bigger environments.

Furthermore, typically, most of the participants did not see a problem with obtaining from other people since this is mostly a matter of contacting the right people. This, however, is different for environments where there is collaboration with multiple external stakeholders in which it becomes increasingly difficult to gain more specific information when there is a need to establish traceability.

Managed

Looking at the results of all the participants collectively, most of the participants indicated that traceability is currently mostly performed in an ad-hoc and non-managed fashion and most of the participants also indicated that there is a need to perform traceability in a more managed fashion.

Of the three SDLC groups, over half of the participants in the Traditional SDLC indicated that traceability is not mostly performed in an ad-hoc and non managed version, while the opposite was true for Agile and Mixed SDLCs. Furthermore, over half of the participants in all three groups indicated that there is a need to perform traceability in a more managed fashion.

Furthermore, over half of the participants in the safety critical projects indicated that traceability is mostly not performed in an ad-hoc and non-managed fashion while this was the case in non-safety critical projects.

Based on the interview results, the answers to the statements can vary based on the artifacts that are traced and the setup of the tooling itself. In addition, it can differ depending on the way of working per team and depending on how the responsibility of traceability is given, e.g. if people are responsible to do it on their own or if there are specific people that focus on it as their main task. Furthermore, it is also notable that even though traceability can be a more managed process, it might not be enforced in which people are still left with the choice to do it or not. And that it will depend on the motivation of the people to do it or not, which could for example be motivated in general by the area in which they work where safety and transparency can be important or just motivated by the benefits themselves.

The findings in this study are in line with previous findings and discussions from studies such as Cleland-Huang et al., (2014, 2012); Mäder et al., (2009); Ramesh, (1998); and Regan et al., (2012), in which many of the participants reported to be performing traceability in an ad hoc and non-managed fashion. In addition, we also see that in Traditional and or safety critical projects traceability is more managed compared to the non-Traditional and non-safety critical projects. Furthermore, even if traceability is managed, it might not be enforced and the degree on how managed traceability is and the links that are established also depends on the motivation and discipline of people.

Different stakeholder viewpoints

Collectively, most of the participants were in agreement that traceability is of high importance for the software development process, and that over half of the participants indicated that there is a need to increase the awareness of the importance of traceability.

When comparing the three SDLC groups, all over half of the participants of the three groups indicated that that traceability is of high importance for the software development process, and also that there is a need to increase the awareness of the importance of traceability.

From both project types, most of the participants agreed that traceability is of high importance and that there is a need to increase the awareness of the importance of traceability.

According to the interviews, we see that most of the answers logically depends on the goals and tasks of the participants and if traceability is part of their main job or if they perceive that it supports their job. Most of our participants see the importance of traceability, even though not all benefits might be clear. However, based on the interviews it became clear that there are many mixed opinions in regard to traceability and that awareness of the benefits and or the concept as a whole plays a big part.

The findings in regards to the importance of traceability and the awareness support the topics and findings as discussed in the studies of Bouillon et al., (2013); Cleland-Huang et al., (2012); Gotel et al., (2012); Ramesh, (1998), and Regan et al., (2012). Similar to these studies, most of the participants indicated the traceability is important for the software development process. However, the participants also indicated that there are many mixed views regarding traceability and that people are also not aware of traceability in their companies.

Allocation of resources

Regarding the topic of allocation of resources, most of the participants indicated that the allocation of time, staff, and resources are often insufficient to be able to properly establish and maintain traceability, and that over half of the participants prioritized the need for more staff, time, and resources to properly establish and maintain traceability of no need or low priority.

When comparing the three SDLC groups, all over half of the participants of the three groups indicated that that the allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability. However, over half of the participants in the Agile group indicated that the need for more staff, time, and resources to properly establish traceability and maintain traceability is of low or no need. This is in contrast to the Traditional group in which over half of the participants indicated a need, while the mixed group were split about the statement

Furthermore, most participants of both project types indicated that the allocation of time, staff, and resources are often insufficient. Nonetheless, over half of the participants in the non-safety projects indicated that there is low or no need for more staff, time, and resources. In case of safety critical projects they were split evenly.

Based on the interviews we see that the time, resources, and staff are related to each other, in which the one affects the other. However, when there is a shortage it is usually directed to not having enough staff. Furthermore, even if there is enough staff or if more time, resources, and staff would be thrown to solve the lack thereof, it does not mean the problem the lack of traceability is fixed since it also depends on the people themselves and their motivations. In addition, having enough traceability and or no need for more time, resources and staff is also attributed to having the right tooling that provides traceability.

The findings in regards to resource allocation also support the findings similar to the studies of Gotel & Finkelstein, (1994), and Ramesh, (1998), most of the participants indicated that there are often insufficient resources to establish traceability but that the problem of insufficient resources is of low priority. In addition, we also see that the shortage is mostly direct to having not enough staff to conduct traceability practices.

Collaboration

Over half of the participants agreed that there is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability, and over half of the participants also indicated that there is a need for more collaboration between involved stakeholder teams regarding traceability.

When comparing the SDLC groups, over half of the participants in the Agile and Traditional group indicated that there is a lack of collaboration in contrast to the mixed SDLC group. In addition, of halve of the participants in all three groups agreed that there is a need for more collaboration between involved stakeholder teams regarding traceability.

Furthermore, most of the participants in the safety critical projects indicated that there is a lack of collaboration between involved stakeholder teams in regards to establishing and maintain traceability while this was not the case for non-safety critical projects. Additionally, most of the participants of both project types indicated that there is a need for more collaboration.

Based on the interviews, the answers can vary based on the size and structure of the companies. In smaller companies there is less need for formal collaboration and communication in regards to Traceability compared to larger and more structured companies. Furthermore, in cases where there are practices such as team meetings or any form of collaboration it is also mainly to align everyone on what they are doing.

Most of the participants indicated that there is a lack of collaboration and that there is a need for more collaboration. Similar to the findings of Wohlrab et al., (2018), we also see differences between underlying development paradigms but also between safety-critical and non-safety critical project types.

Responsibility

Over half of the participants indicated that it is unclear for them which roles are responsible for traceability, and most of the participants indicated that there is a need for more communication about who are responsible for traceability.

Over half of the participants of each of the three groups agreed that it is unclear which roles are responsible for traceability and that there is a need for more communication about who are responsible for traceability.

Regarding project types, over half of the participants in non-safety critical projects indicated that it is unclear which roles are responsible for traceability, while it was mixed between safety-critical projects. Additionally, over half of the participants in both groups indicated that there is a need for more communication about who are responsible for traceability.

According to the interviews, we see that in cases where the responsibilities of which roles perform traceability are not clear, is in cases where the responsibilities of the roles themselves are also not clear. This could be due to being in a flexible environment such as startups where everyone does a bit of everything, or it could be in the cases where the responsibilities of multiple overlap each other. In addition, this seems to be also linked to cultural aspects, where in some cultures there is more of a do vs a more planning mentality.

Similar to the topics and findings discussed in the studies of Gotel & Finkelstein, (1994), and Regan et al., (2012), we also see that it is unclear for most participants which roles are responsible for traceability and we also see that there is a need for more communication regarding this topic.

Tools

Just over half of the participants indicated that traceability tools do not satisfy their traceability needs and just over half of the participants indicated that there is a low need or no need for better traceability tools. However, on both statements of their situation and needs, the participants were fairly evenly divided.

The results of comparing the three SDLC groups, showed that for the Agile SDLC group the traceability tools satisfy their traceability needs, while this was not the case for both mixed and Traditional SDLC groups. Furthermore, over half of the participants in the mixed group indicated that there is no need for better traceability tools compared to the other groups.

In relation to the project type, over half of the participants in the safety critical projects indicated that the traceability tools do not satisfy their need, while the tools do satisfy the need for over half of the participants in non-safety critical projects. In addition, for over half of the participants in non-safety critical projects there was no or low need for better traceability tools, while the participants in the safety critical projects were split.

Based on the interviews, in most cases of our participants, the tools that are used satisfy their main needs and tasks. However, most of the needs seem more related to the quality aspects in which there is an emphasis on reducing effort as to inputting trace links and linking different artifacts and or different tools to each other, which could provide a better and overall traceability picture.

The results regarding the topic of tools is similar to the topics and findings as discussed in the studies of Gotel et al., (2012), and Mäder et al., (2009). Most of the participants were not satisfied with their traceability tools however there was a low priority for better or new tools. In multiple cases, the need was more directed in improving their current tools that were already in use since most of the essential traceability functions were possible. In addition, we also perceived that even if the tools had all the functionalities that the participants needed, these functionalities were not always apparent.

Storage and versioning

Similar to the previous topic, the participants were evenly divided as to it is difficult to establish traceability because development artifacts are stored in multiple location/repositories, but more than half of the participants indicated that the need for a more centralized development artifact repository to establish traceability more easily is of low priority or no need.

For the Traditional SDLC group, over half of them indicated that it is difficult to establish traceability because development artifacts are stored in multiple locations/repositories compared to the Agile and mixed SDLC groups. Additionally, over half of the participants in the Traditional group indicated that there is a need for a more centralized development artifact repository to establish traceability more easily compared to the Agile and Mixed SDLC groups.

Regarding storage and versioning for project types, over half of the participants in non-safety critical projects indicated that it is difficult to establish traceability because development artifacts are stored in multiple locations/repositories while the opposite was the case for safety critical projects. In addition, most of the participants of both project types indicated that there was no need for a more centralized development artifact repository to establish traceability more easily.

Reasons for disagreeing is in most cases also straightforward in which most of the artifacts are stored centrally in tools or where they make use of tool suites that easily connect to each other. In cases where it was difficult to establish traceability it was mainly because of usage of a variety of tools and lack of interfaces as described in the previous topic.

The results support the findings and topics discussed in studies from Gotel & Finkelstein, (1994); Ramesh, (1998), and Regan et al., (2012). Nonetheless, the results also indicate that the need for a more centralized repository is of low priority or no need. However, the difficulty and need increases when multiple tools are in play and in bigger environments.

Complexity

More than half of the participants indicated that the tools used for traceability are not too complex to use effectively and to integrate with their existing tools, while just over half of the participants

indicated that there is a need for less complex traceability tools and easier integration with their existing tools.

Over half of the participants of all three groups indicated that the tools used for traceability are not too complex to use and effectively integrate with their existing tools. Nonetheless, over half of the participants in the mixed group indicated that there is a low or no need for less complex traceability tools and easier integration with their existing tools in contrast to the Agile and Traditional SDLC groups.

Both project types indicated that the traceability tools were not too complex to use and integrate with their existing tools. Nonetheless the participants in the safety critical projects were split, while there was also a need in non-safety critical projects for less complex traceability tools and easier integration with their existing tools.

The consensus was that most tools are not too complex to use and to integrate with their existing tools. Reasons given was that in several cases, tools of a similar suite were used and therefore of course there are no problems with integrating tools. In addition, in some cases traceability was just a byproduct of the main tasks they were performing in the tools which they were familiar with in which there is no need for less complex tools. In some cases, where participants agreed with the statement, it could be due to lack of awareness of the features in their main tools and the use of multiple tools which added to the complexity.

Automation
Many of the participants indicated that traceability is mostly performed manually, and over half of the participants indicated that there is a need for more traceability automation.

Over half of the participants of all three groups indicated that traceability is mostly performed manually and that there is a need for more traceability automation.

The same findings were found when comparing project types, in which all two groups in safety critical and non-safety critical indicated that traceability is mostly performed manually, and that there is a need for more traceability automation.

Based on this, reasons for performing traceability manually and having a need for more automation could be because it is part of the tools and the way of working of these tools. Other reasons also because of the low recall and precisions of automated tools that have been tested and are not yet on a level to be used in an actual setting.

These finding regarding the topic of traceability automation indicated to support the research efforts conducted by the software traceability research community. Currently, many of the interviewed participants still create and maintain their traces manually. This is not a problem for smaller number of traces; however, it becomes a problem when having to deal with traces in the order of 10s of thousands. In addition, even though part of the participants think that a fully automated approach will not be possible or is not suited for their environment, they would still like to see their efforts made to create and maintain trace links reduced by support of automation.

9.3 Limitations

There are multiple limitations of this study that has to be taken into account that can affect this study. The limitations are structured and discussed according conclusion validity, internal validity, construct validity, and external validity.

Conclusion validity is concerned with the relationship between treatment and outcome, in other words it is concerned with the degree how reasonable the conclusions, relations, and inductions derived from this study are compared to the data.

One of the threats that is inherent to this study is the random heterogeneity of respondents. Because the study was set up to reach as many respondents as possible via the questionnaire, it also means that there is a diverse group of respondents. As consequence, the conclusions, relations, and inductions made based on the data might not reflect certain subsets of population to a certain extent. However, the goal of the questionnaire was to gather data from as many people as possible in varying roles.

Another threat related to conclusion validity, is the possible statistical power of the analyses used on the data. For this, we made sure that the proper statistical tests were used and that the right assumptions were satisfied which lead to the use of non-parametric tests compared to parametric tests. The downside, however, is that typically proper parametric tests have more statistical power. Furthermore, with a sample size of 55, the sample might just be adequate for drawing general conclusions based on the whole sample size. However, because of this sample size, the story is different when comparing and subdividing the data into multiple categories. The most prominent example of this threat is that the paradigm category of 'Traditional' SDLC was not nearly as saturated compared to the other two groups of Agile and Mixed. Because of this, the conclusions, relations, and inductions made based on this group are under threat of low statistical power. This also means that the significant difference found between Agile, and Traditional, regarding performing traceability manually, might be in the realm of a type 1 error, in which a relationship was found while there in fact might not be one. Furthermore, in the cases of where there were no significant differences found, these results might be subject to type 2 errors. Therefore, despite the efforts of increasing the sample size and therefore also the statistical power by putting the questionnaire out for a longer time then intended and also personally contacting people, because of the used sample size, we have to acknowledge the threat of a low statistical power in the study.

Internal validity considers how the experiment minimizes bias

A threat of performing a survey is finding a representative group. This is not a problem to a certain extend considering we were interested in finding practitioners that were active and or interested in the topic. To do so, we decided to perform an online survey, which offered the opportunity for a wide population of practitioners to participate, in which the survey was advertised across multiple social platforms. However, this included the risk of only inviting practitioners that are active and or interested in the topic of software traceability, as well illegible people that do not meet the target demographic. In order to partially mitigate that threat, we stated in the introductory page of the questionnaire the requirements of being eligible to fill in the survey. Furthermore, we also took efforts to advertise our study through personal contacts. However, it is therefore also important to note that there is a potential threat of bias considering that it is also possible that, even if practitioners are active in the topic, they might not have interest in traceability and would therefore not fill in the questionnaire.

Another form of bias that might be a threat is that the current survey design does not allow us to check if participants are from different companies or the same company. Even though, this is not really an issue considering that the survey is about the perceptions of practitioners themselves and that multiple participants have mentioned that there are mixed perceptions in their company. This does mean that multiple participants from the same company could have filled in the questionnaire and

that there might be bias based on the assumption that multiple practitioners from the same company could have the same perceptions on traceability because they are in the same environment.

Furthermore, regarding the interviews, the participants were chosen based on that fact that they have filled in the questionnaire and indicated in the survey that they were willing to partake in a follow up interview. Because not every participant from the questionnaire could be interviewed, it is possible that the findings of the interviews are biased to just a subset of the participants. Nonetheless, we did see from the people that were interviewed that most of them were diverse enough form each other to be able to induct general findings to a certain extend and to partially prevent bias in the interview results based on the interviewees.

Construct validity involves the relation between theory and observation

By conducting a questionnaire-based survey, we tried to eliminate the influence of the experimenter on the subject as far as possible. We aimed for a simple language for our questions to partially mitigate ambiguity which is also a risk of questionnaire based surveys, and we tested this, including the structure of the questionnaire, with multiple feedback sessions with other master students and a native English speaker from the UK. In addition, we also validated the content with a practitioner to gain feedback on if the questions are understandable from a practitioner's view and on potential missing information. Ideally, we would have liked to test the survey with multiple practitioners, however this was not possible due to time and resource constraints. Despite these efforts, it is possible that some of the questions and statements suffer from ambiguity and could still be improved and refined. Nonetheless, based on the results of the questionnaire, almost all the participants managed to fill in most of the questions.

Furthermore, regarding the interviews, we tried to eliminate the influence of the experimenter partially as well by performing a semi-structured interview in which the structure was based on the questionnaire. Nonetheless, even though the interview questions were asked as neutral as possible, considering that it was semi-structured, we cannot exclude the possibility that the interview might have influenced the subjects partially during the discussions of the rationale and answers of the subjects. In addition, there is also a potential threat of misunderstanding the participants. Furthermore, bias can also potentially be introduced during the coding of the interviews. To eliminate the bias of the experimenter partially, the codes were structured consistently with the questionnaire and interview structured. However, it is important to note that these threats are inherent to most qualitative studies.

External validity threats reduce generalization of the findings

Our sample consisted of 55 participants. We were able to gather data from a diverse group of participants. Nonetheless, we do see that most of that participants are from the Netherlands, are from large companies, and most of them work in an Agile SDLC environment. This indicated that, even though we have a diverse group of participants to a certain extent, that the findings can only be generalized to a certain point.

Furthermore, based on the interviews, we did see that we succeeded in gaining the perceptions of practitioners that view traceability differently from each other and are interested in a variety of subsets of traceability, such as pre-requirements traceability and post-requirements traceability. However, even though the findings are generalizable to this extend, it is important to keep in mind that the views and perceptions of participants might change depending on what subset of software traceability is discussed and this might limit the generalizability of the results.

10. Conclusion and Future directions

In conclusion, based on the data gathered from 55 participants, we only found one significant difference in the perceived value of software traceability between Agile, and non-Agile SDLCs, which was about if software traceability is mostly performed manually or not. Besides that, we did not find any significant differences between Agile and non-Agile SDLCs as well as between safety critical and non-safety critical projects.

Even though this study found one significant difference, it did find a few differences regarding the sentiment between these groups, and also which statements were agreed on the most and what is needed the most.

Nonetheless, despite the smaller differences found regarding the current traceability situation and needs, the results indicate and seem to suggest that, regardless of which SDLC and project type is followed, similar traceability situations are perceived, such as that Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices, that traceability is of high importance for the software development process, and that traceability is mostly performed manually. In addition, despite small differences in the rankings, similar needs are perceived as high priority as well, such as that there is a need to perform traceability in a more managed fashion, that there is a need for a more clear overview of the costs and benefits regarding traceability, that there is a need for more traceability automation, that there is a need to increase the awareness of the importance of traceability, and that there is a need for more collaboration and guidance regarding traceability.

As such, even though there are smaller differences in the sentiment of the participants, overall, the findings seem to suggest that similar traceability issues, perceptions, and needs, are found regardless of which SDLC is followed and if the project is safety critical or not, despite the underlying differences between these groups.

Furthermore, based on the collective sentiment of all the participants and the interviews, we also perceived many of the topics and problems that were found, are currently being researched and supported by the traceability research community. In addition, more insight is created by this study as to which of the topics and problems are prioritized more than the others.

Therefore, by having conducted this study, we see contributions to the need to gain more insight regarding software traceability in practice between the typical domains and non-typical domains, as well as to the general body of knowledge of software traceability in practice, strengthened and validated the findings of previous empirical traceability work, and also collected data for future studies.

However, much more work should be conducted in the field of software traceability to gain even more insight into the problems and needs encountered in practice.

Based on the observations of this study we see multiple directions for future work. First, a follow up study including a similar questionnaire can be conducted to verify and validate the observations found in this study. This follow up study could include a larger sample size, improvements based on the feedback of all the participants that filled in the questionnaire, and additional improvements based on the knowledge gained of conducting this kind of study in the context of software traceability. Secondly, another follow-up study that can be done is regarding eliciting more information on the rationale of the practitioners regarding their current traceability situations and their needs in a variety of environments and tool setups. Thirdly, additional work can be conducted with more specific constructs and goals such as a study focusing only on the different SDLC methods or the different

project types such as safety critical and non-safety critical projects. Fourthly, considering there are many types of software traceability and artifacts, studies focusing on different subtypes of software traceability would also serve to gain more insight in which types of traceability are valued in practice. Fifthly, considering that we did not analyze every demographic factor in this study, smaller follow up studies can also be conducted based on the dataset gathered in which other variables and demographic factors can be sub divided to gain insight into how the perceptions of the practitioners change and the influence on the findings as reported in this study. Finally, even though this study gathered empirical evidence regarding practitioners, we still think it is important to conduct more work in this area considering there is still need for more empirical evidence regarding software traceability. In addition, the interviewed participants in this study showed appreciation and indicated that it would be beneficial to conduct more of these types of studies.

Bibliography

{DO-178C/ED12C} --- Software Considerations in Airborne Systems and Equipment Certification. (2012).

Agility Alliance, A. (2011). Manifesto for Agile Software Development, www.agilemanifesto.org. Retrieved from http://academic.brooklyn.cuny.edu/cis/sfleisher/Chapter_03_sim.pdf

Alsalemi, A. M., & Yeoh, E. T. (2016). A survey on product backlog change management and requirement traceability in agile (Scrum). *2015 9th Malaysian Software Engineering Conference, MySEC 2015*, 189–194. https://doi.org/10.1109/MySEC.2015.7475219

Antoniol, G., Cleland-Huang, J., Hayes, J. H., & Vierhauser, M. (2017). Grand Challenges of Traceability: The Next Ten Years. Retrieved from http://arxiv.org/abs/1710.03129

Arkley, P., & Riddle, S. (2005). Overcoming the traceability benefit problem, 385–389. https://doi.org/10.1109/re.2005.49

Basili, V. R., Caldiera, G., & Rombach, H. D. (1994). The goal question metric approach. *Encyclopedia of Software Engineering*, 528–532. Retrieved from http://fub-taslim.googlecode.com/svn/trunk/WEMSE/INSTICC_Conference_Latex/gqm.pdf

Blaauboer, F., Sikkel, K., & Aydin, M. N. (2007). Deciding to Adopt Requirements Traceability in Practice, 294–308. https://doi.org/10.1007/978-3-540-72988-4_21

Borg, M., Runeson, P., & Ardö, A. (2014). Recovering from a decade: a systematic mapping of information retrieval approaches to software traceability. *Empirical Software Engineering*, *19*(6), 1565–1616. https://doi.org/10.1007/s10664-013-9255-y

Bouillon, E., Mäder, P., & Philippow, I. (2013). A survey on usage scenarios for requirements traceability in practice. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7830 LNCS*, 158–173. https://doi.org/10.1007/978-3-642-37422-7_12

Cleland-Huang, J. (2006). Just enough requirements traceability. *Proceedings - International Computer Software and Applications Conference*, *1*, 41–42. https://doi.org/10.1109/COMPSAC.2006.57

Cleland-Huang, J. (2012). Traceability in agile projects. *Software and Systems Traceability*, *9781447122*, 265–275. https://doi.org/10.1007/978-1-4471-2239-5_12

Cleland-Huang, J., Gotel, O. O. C. Z., Huffman Hayes, J., Mäder, P., Zisman, A., Hayes, J. H., … Zisman, A. (2014). Software Traceability: Trends and Future Directions. *Proceedings of the on Future of Software Engineering*, 55–69.

Cleland-Huang, J., Zisman, A., & Gotel, O. (2012). Software and systems traceability. *Software and Systems Traceability*, *9781447122*, 1–491. https://doi.org/10.1007/978-1-4471-2239-5

COLLAB.NET, & VERSIONONE.COM. (2018). Versionone 12th annual State of Agile Report. *State of Agile*, 16. https://doi.org/10.1080/13557858.2016.1246518

Dekhtyar, A., Poly, C., Obispo, S. L., & Hayes, J. H. (2018). Automating Requirements Traceability :

Two Decades of Learning from KDD, 12–15. https://doi.org/10.1109/D4RE.2018.00009
Egyed, A., Grunbacher, P., Heindl, M., & Biffl, S. (2007). Value-based requirements traceability: lessons learned.
Espinoza, A., & Garbajosa, J. (2011). A study to support agile methods more effectively through traceability. *Innovations in Systems and Software Engineering*, *7*(1), 53–69. https://doi.org/10.1007/s11334-011-0144-5
European Commision. (2015). *User guide to the SME Definition*. *User guide to the SME Definition*.
Fernandez, R. S., & Bonillo, M. A. I. (2007). The concept of perceived value: a systematic review of the research. *Marketing Theory*, *7*(4), 427–451. https://doi.org/10.1177/1470593107083165
Furtado, F., & Zisman, A. (2016). Trace++: A Traceability Approach to Support Transitioning to Agile Software Engineering. *Proceedings - 2016 IEEE 24th International Requirements Engineering Conference, RE 2016*, 66–75. https://doi.org/10.1109/RE.2016.47
Gotel, O. C. Z., & Finkelstein, C. W. (1994). An analysis of the requirements traceability problem. *Proceedings of IEEE International Conference on Requirements Engineering*, 94–101. https://doi.org/10.1109/ICRE.1994.292398
Gotel, O., Cleland-Huang, J., Hayes, J. H., Zisman, A., Egyed, A., Grunbacher, P., & Antoniol, G. (2012). The quest for Ubiquity: A roadmap for software and systems traceability research. *2012 20th IEEE International Requirements Engineering Conference, RE 2012 - Proceedings*, 71–80. https://doi.org/10.1109/RE.2012.6345841
Gotel, O., Cleland-Huang, J., Hayes, J. H., Zisman, A., Egyed, A., Grünbacher, P., … Maletic, J. (2012). The grand challenge of traceability (v1.0). *Software and Systems Traceability*, *9781447122*, 343–409. https://doi.org/10.1007/978-1-4471-2239-5_16
Grand Challenges of Traceability 2017. (n.d.).
Guo, J., Cheng, J., & Cleland-Huang, J. (2017). Semantically Enhanced Software Traceability Using Deep Learning Techniques. *Proceedings - 2017 IEEE/ACM 39th International Conference on Software Engineering, ICSE 2017*, 3–14. https://doi.org/10.1109/ICSE.2017.9
Ingram, C., & Riddle, S. (2013). Cost-benefits of traceability. *Software and Systems Traceability*, *9781447122*, 23–42. https://doi.org/10.1007/978-1-4471-2239-5_2
K. Petersen, R. Feldt, S. M. (2008). Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering* (Vol. 8, p. 1–10.). https://doi.org/10.1142/S0218194007003112
Kitchenham, B. (2004). Procedures for performing systematic reviews. *Joint Technical Report, Computer Science Department, Keele University*, (TR/SE-0401), 28. https://doi.org/10.1.1.122.3308
Knight, J. C. (2003). Safety critical systems: challenges and directions, 547–550. https://doi.org/10.1109/icse.2002.1007998
Lam, A. N., Nguyen, A. T., Nguyen, H. A., & Nguyen, T. N. (2016). Combining deep learning with information retrieval to localize buggy files for bug reports. *Proceedings - 2015 30th IEEE/ACM International Conference on Automated Software Engineering, ASE 2015*, 476–481. https://doi.org/10.1109/ASE.2015.73
Mäder, P., & Egyed, A. (2015). Do developers benefit from requirements traceability when evolving and maintaining a software system? *Empirical Software Engineering*, *20*(2), 413–441. https://doi.org/10.1007/s10664-014-9314-z
Mäder, P., Gotel, O., & Philippow, I. (2009). Motivation matters in the traceability trenches. *Proceedings of the IEEE International Conference on Requirements Engineering*, 143–148. https://doi.org/10.1109/RE.2009.23
Mc Hugh, M., Mc Caffery, F., & Casey, V. (2012). Barriers to using Agile Software Development Practices within the Medical Device Industry. *European Systems and Software Process Improvement and Innovation Conference, EuroSPI 2012*, 0–7. Retrieved from http://ulir.ul.ie/handle/10344/2442
Miranda, E. (2011). Time boxing planning: Buffered MOSCOW rules. *ACM SIGSOFT Software*

Engineering Notes, *36*(6), 1. https://doi.org/10.1145/2047414.2047428
Peterson, R. (2014). *Constructing Effective Questionnaires*. *Constructing Effective Questionnaires*. https://doi.org/10.4135/9781483349022
Pierce, R. A. (1978). A Requirements Tracing Tool. *ACM SIGSOFT Software Engineering Notes*, *3*(5), 53–60. https://doi.org/10.1145/953579.811100
Ramesh, B. (1998). Factors influencing requirements traceability practice. *Communications of the ACM*, *41*(12), 37–44. https://doi.org/10.1145/290133.290147
Regan, G., McCaffery, F., McDaid, K., & Flood, D. (2012). The barriers to traceability and their potential solutions: Towards a reference framework. *Proceedings - 38th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2012*, 319–322. https://doi.org/10.1109/SEAA.2012.80
Rempel, P., & Mader, P. (2017). Preventing defects: The impact of requirements traceability completeness on software quality. *IEEE Transactions on Software Engineering*, *43*(8), 777–797. https://doi.org/10.1109/TSE.2016.2622264
Road vehicles — Functional safety — Part 6: Product development at the software level. (2012), *2011*.
van Solingen Rini, Vic, B., Gianluigi, C., & Dieter, R. H. (2002). Goal Question Metric (GQM) Approach. *Encyclopedia of Software Engineering*. https://doi.org/10.1002/0471028959.sof142
Waters, K. (2009). Prioritization using MoSCoW. *Agile Planning*, *12*, 31.
Wieringa, R. (2014). *Design Science and Software Engineering*. *Design Science Methodology for Information Systems and Software Engineering*. Springer. https://doi.org/10.1007/978-3-662-43839-8
Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering - EASE '14*, 1–10. https://doi.org/10.1145/2601248.2601268
Wohlrab, R., Knauss, E., Steghöfer, J. P., Maro, S., Anjorin, A., & Pelliccione, P. (2018). Collaborative traceability management: a multiple case study from the perspectives of organization, process, and culture. *Requirements Engineering*. https://doi.org/10.1007/s00766-018-0306-1
Zhang, Y., Witte, R., Rilling, J., & Haarslev, V. (2008). Ontological approach for the semantic recovery of traceability links between software artefacts. *IET Software*, *2*(3), 185. https://doi.org/10.1049/iet-sen:20070062

Appendix

Additional resources such as R scripts, the anonymized data set, and other artifacts are made available at: https://drive.google.com/open?id=1jrmvSWUqMiw8tTdc4Pq9d081HaxDOAad

Appendix A: Glossary

This section contains the glossary of the terms used in this study. As described in the introductory sections, this study closely follows the definitions as described by Cleland-Huang et al., (2012). As such, this glossary contains relevant reproduced parts of their glossary from their book Software and Systems traceability. A more extensive and up to date glossary can be found on their website http://www.coest.org. The reproduced parts can be found in the blue table. Other important definitions used throughout this study can be found in the orange table.

Relevant reproduced parts of the glossary of	Cleland-Huang et al., (2012)
Artifact	Something that is created or shaped by humans, either directly or indirectly via automation. In software and systems engineering connects, the term refers to the products of the engineering process.
Atomic trace	A trace compromising a single source artifact, a single target artifact, and a single trace link.
Automated traceability	The potential for automated tracing
Automated tracing	When traceability is established via automated techniques, methods, and tools. Currently, it is the decision as to among which artifacts to create and maintain trace links that is automated.
Backward tracing	In software and systems engineering contexts, the term is commonly used when the tracing follows antecedent steps in a developmental path, which is not necessarily a chronological path, such as backward from code through design to requirements. Note that the trace links themselves could be used in either a primary or reverse trace link direction, dependent upon the specification of the participating traces.
Bidirectional trace link	A term used to refer to the fact that a trace link can be used in both a primary trace link direction and a reverse trace link direction.
Center of Excellence for Software Traceability (CoEST)	A traceability community initiative. “Our goal is to bring together traceability researchers and experts in the field. We hope to encourage research collaborations, assemble a body of knowledge for traceability, and develop new technology to meet tracing needs.” (Hayes et al., 2007.) See: http://www.coest.org.
Chained trace	A trace (noun sense) comprising multiple atomic traces strung in sequence, such that a target artifact for one atomic trace becomes the source artifact for the next atomic trace.
Forward tracing	In software and systems engineering contexts, the term is commonly

	used when the tracing follows subsequent steps in a developmental path, which is not necessarily a chronological path, such as forward from requirements through design to code. Note that the trace links themselves could be used in either a primary or reverse trace link direction, dependent upon the specification of the participating traces.
Grand Challenge of Traceability	A fundamental problem with traceability that members of the international research and industrial communities agree deserves attention in order to achieve a revolutionary advance in traceability practice. It is a problem with no point solution; its solution involves first understanding and tackling a myriad of underlying challenges, and so will demand the effort of multiple research groups over an extended time period.
Horizontal tracing	In software and systems engineering contexts, the term is commonly used when tracing artifacts at the same level of abstraction, such as: (i) traces between all the requirements created by "Mary", (ii) traces between requirements that are concerned with the performance of the system, or (iii) traces between versions of a particular requirement at different moments in time. Horizontal tracing may employ both forward tracing and backward tracing.
Just in time tracing (JITT)	See reactive tracing.
Link semantics	The purpose or meaning of the trace link. The link semantics are generally specified in the trace link type, which is a broader term that may also capture other details regarding the nature of the trace link, such as how the trace link was created.
Manual tracing	When traceability is established by the activities of a human tracer. This includes traceability creation and maintenance using the drag and drop methods that are commonly found in current requirements management tools.
Post-requirements (specification) tracing	In software and systems engineering contexts, the term is commonly used to refer to those traces derived from or grounded in the requirements, and hence the traceability explicates the requirements' deployment process. The tracing is, therefore, forward from requirements and back to requirements. Post-requirements (specification) tracing may employ forward tracing, backward tracing, horizontal tracing and vertical tracing.
Pre-requirements (specification) traceability	In software and systems engineering contexts, the term is commonly used to refer to those traces that show the derivation of the requirements from their original sources, and hence the traceability explicates the requirements' production process. The tracing is, therefore, forward to requirements and back from requirements. Pre-requirements (specification) tracing may employ forward tracing, backward tracing, horizontal tracing and vertical tracing.

Primary trace link direction	When a trace link is traversed from its specified source artifact to its specified target artifact, it is being used in the primary direction as specified. Where link semantics are provided, they provide for a way to "read" the traversal (e.g., A implements B).
Proactive tracing	Initiating trace capture without explicit response to a stimulus to do so (i.e., traces are created in the background). Compare with reactive tracing.
Reactive tracing	Responding to a stimulus to initiate trace capture (i.e., traces are created on demand). Compare with proactive tracing.
Requirements management	The activity concerned with the effective control of information related to stakeholder, system and software requirements and the preservation of the integrity of that information for the life of the system and with respect to changes in the system and its environment. Requirements management depends upon requirements traceability as its enabling mechanism.
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 periods of ongoing refinement and iteration in any of these phases)." (Gotel and Finkelstein, 1994.)
Semi-automated tracing	When traceability is established via a combination of automated techniques, methods, tools and human activities. For example, automated techniques may suggest candidate trace links or suspect trace links and then the human tracer may be prompted to verify them.
Software traceability	See requirements traceability, extending the definition to encompass and interrelate any uniquely identifiable software engineering artifact to any other.
Source artifact	The artifact from which a trace originates.
Systems traceability	See requirements traceability, extending the definition to encompass and interrelate any uniquely identifiable systems engineering artifact to a broad range of systems-level components, such as people, processes and hardware models.
Trace (Noun)	A specified triplet of elements comprising: a source artifact, a target artifact and a trace link associating the two artifacts. Where more than two artifacts are associated by a trace link, such as the aggregation of two artifacts linked to a third artifact, the aggregated artifacts are treated as a single trace artifact. The term applies, more generally, to both traces that are atomic in nature (i.e., singular) or chained in some way (i.e., plural).
Trace (Verb)	The act of following a trace link from a source artifact to a target artifact (primary trace link direction) or vice-versa (reverse trace link direction). See tracing.

Trace generation	A particular approach to trace creation that implies that the trace links are created automatically or semi-automatically using tools.
Trace granularity	The level of detail at which a trace is recorded and performed. The granularity of a trace is defined by the granularity of the source artifact and the target artifact.
Trace life cycle	A conceptual model that describes the series of activities involved in the life of a single trace, from initial conception, through creation, maintenance and use, through to eventual retirement. This is the traceability process from the perspective of a single trace flowing through the traceability process.
Trace link	A specified association between a pair of artifacts, one comprising the source artifact and one comprising the target artifact. The trace link is one of the trace elements. It may or may not be annotated to include information such as the link type and other semantic attributes. This definition of trace link implies that the link has a primary trace link direction for tracing. In practice, every trace link can be traversed in two directions (i.e., if A tests B then B is tested by A), so the link also has a reverse trace link direction for tracing. The trace link is effectively bidirectional. Where no concept of directionality is given or implied, it is referred to solely as an association.
Trace link type	A label that characterizes those trace links that have the same or similar structure (syntax) and/or purpose (semantics). For example, “implements”, “tests”, “refines” and “replaces” may be distinct trace link types.
Trace maintenance	Those activities associated with updating a single preexisting trace as changes are made to the traced artifacts and the traceability evolves, creating new traces where needed to keep the traceability relevant and up to date.
Trace relation	All the trace links created between two sets of specified trace artifact types. The trace relation is the instantiation of the trace relationship and hence is a collection of traces. For example, the trace relation would be the actual trace links that associate the instances of requirements artifacts with the instances of test case artifacts on a project. The trace relation is commonly recorded within a traceability matrix.
Trace relationship	An abstract definition of a permissible trace relation on a project (i.e., source artifact type, target artifact type and trace link types), as typically expressed within a traceability information model (TIM). Note that the trace links of the instances of the two artifact types may not necessarily have the same trace link type.
Trace retrieval	A particular approach to trace creation where information retrieval methods are used to dynamically create a trace link. This approach can be used for both trace capture and trace recovery.

Trace use	Those activities associated with putting a single trace to use to support various software and systems engineering activities and tasks.
Traceability	The potential for traces to be established and used. Traceability (i.e., trace "ability") is thereby an attribute of an artifact or of a collection of artifacts. Where there is traceability, tracing can be undertaken, and the specified artifacts should be traceable.
Traceability Body of Knowledge (TBOK)	A proposed resource for the traceability community, containing traceability benchmarks, good traceability practices, traceability experience reports, etc.
Traceability challenge	A significant problem with traceability that members of the international research and industrial communities agree deserves attention in order to achieve advances in traceability practice.
Traceability community	Those people who are establishing and using traceability in practice or have done so in the past or intend to do so in the future. Also, those people who are active in traceability research or in one of its many interrelated areas.
Traceability creation	The general activity of associating two (or more) artifacts, by providing trace links between them, for tracing purposes. Note that this could be done manually, automatically or semi-automatically, and additional annotations can be provided as desired to characterize attributes of the traces.
Traceability-enabled activities and tasks	Those software and systems engineering activities and tasks that traceability supports, such as verification and validation, impact analysis and change management.
Traceability evolution	The gradual change of the traceability on a project. It generally, refers to the tendency for pre-existing traces to become outdated and/or obsolete over time as changes are made to the traced artifacts, unless the traceability is maintained sufficiently. Ongoing deterioration of the traceability may lead to traceability decay.
Traceability information	Any traceability-related data, such as traceability information models, trace artifacts, trace links and other traceability work products.
Traceability information model (TIM)	A graph defining the permissible trace artifact types, the permissible trace link types and the permissible trace relationships on a project, in order to address the anticipated traceability-related queries and traceability-enabled activities and tasks. The TIM is an abstract expression of the intended traceability for a project. The TIM may also capture additional information such as: the cardinality of the trace artifacts associated through a trace link, the primary trace link direction, the purpose of the trace link (i.e., the link semantics), the location of the trace artifacts, the tracer responsible for creating and maintaining the trace link, etc. (See Mäder et al. (2009) for more

	detail.)
Traceability maintenance	Those activities associated with updating preexisting traces as changes are made to the traced artifacts and the traceability evolves, creating new traces where needed to keep the traceability relevant and up to date.
Traceability management	Those activities associated with providing the control necessary to keep the stakeholder and system requirements for traceability and the traceability solution up to date during the life of a project. Traceability management is a fundamental part of traceability strategy.
Traceability matrix	A matrix recording the traces comprising a trace relation, showing which pairs of trace artifacts are associated via trace links.
Traceability method	A prescription of how to perform a collection of traceability practices, integrating traceability techniques with guidance as to their application and sequencing.
Traceability planning	Those activities associated with determining the stakeholder and system requirements for traceability and designing a suitable traceability solution. Traceability planning is a fundamental part of traceability strategy.
Traceability process	An instance of a traceability process model defining the series of activities to be employed to establish traceability and render it usable for a particular project, along with a description of the responsibilities and resourcing required to undertake them, as well as their inputs and outputs. The traceability process defines how to undertake traceability strategy, traceability creation, traceability maintenance and traceability use.
Traceability stakeholders	Those roles (i.e., people or systems) that have something to gain or something to lose from either having or not having traceability on a project.
Traceability standard	Mandatory practices and other conventions employed and enforced to prescribe a disciplined and uniform approach to traceability, generally written down and formed by consensus.
Traceability strategy	Those decisions made in order to determine the stakeholder and system requirements for traceability and to design a suitable traceability solution, and for providing the control necessary to keep these requirements and solutions relevant and effective during the life of a project. Traceability strategy comprises traceability planning and traceability management activities.
Traceability technique	A prescription of how to perform a single traceability practice, such as traceability creation, along with a description of how to represent its traceability work products.
Traceability tool	Any instrument or device that serves to assist or automate any

	part of the traceability process.
Traceability use	Those activities associated with putting traces to use to support various software and systems engineering activities and tasks, such as verification and validation, impact analysis and change management.
Value-based traceability	An approach to traceability that actively seeks to create, manage and measure either the monetary worth or utility worth of traceability on a project.
Vertical tracing	In software and systems engineering contexts, the term is commonly used when tracing artifacts at differing levels of abstraction so as to accommodate life cycle-wide or end-to-end traceability, such as from requirements to code. Vertical tracing may employ both forward tracing and backward tracing.

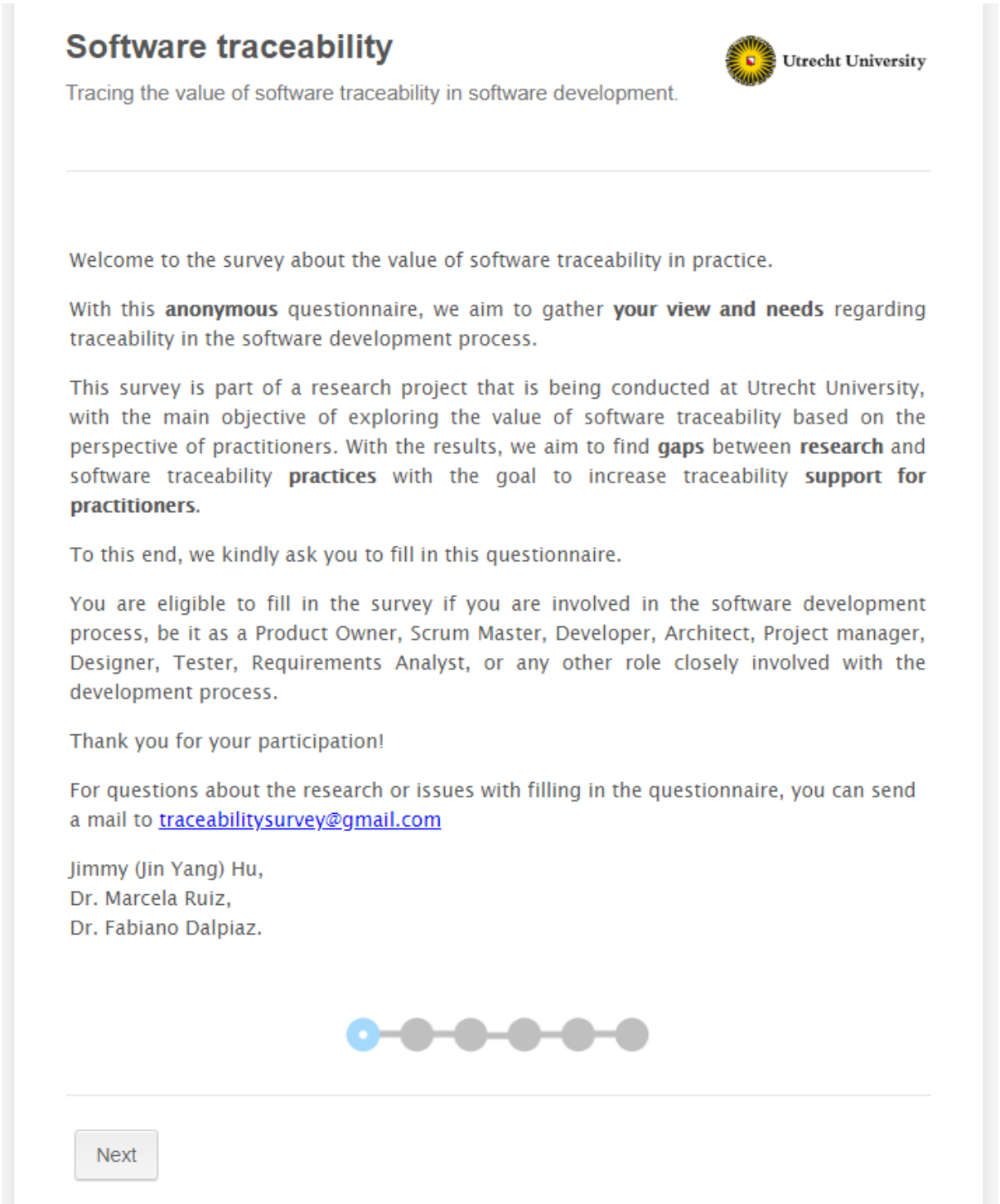
Other important definitions used in this study	
Agile Development	Agile software development is an umbrella term for a set of frameworks and practices based on the values and principles expressed in the Manifesto for Agile Software Development and the 12 Principles behind it. (https://www.agilealliance.org/agile101/)
Benefits	An advantage or profit gained from something.
Costs	An amount that has to be paid or given up in order to get something. In business, cost is usually a monetary valuation of (1) effort, (2) material, (3) resources, (4) time and utilities consumed, (5) risks incurred, and (6) opportunity forgone in production and delivery of a good or service. All expenses are costs, but not all costs (such as those incurred in acquisition of an income-generating asset) are expenses. (http://www.businessdictionary.com/definition/cost.html)
Domain	See Industry
Hedonic	Relating to, characterized by, or considered in terms of pleasant (or unpleasant) sensations
Industry	Industry refers to a specific group of companies that operate in a similar business sphere. Essentially, industries are created by breaking down sectors into more defined groupings. Therefore, these companies are divided into more specific groups than sectors. Each of the dozen or so sectors will have a varying number of industries, but it can be in the hundreds. (https://www.investopedia.com/ask/answers/05/industrysector.asp)
Non-safety critical Projects	Projects that develop and maintain non-safety critical systems

Non-safety critical systems	Opposite of safety-critical systems
Perceived Value	the value of a product based on how much customers want or need it, rather than on its real price. (https://dictionary.cambridge.org/dictionary/english/perceived-value)
Safety-critical projects	Projects that develop and maintain a safety-critical system
Safety-critical systems/software	Safety-critical systems are those systems whose failure could result in loss of life, significant property damage or damage to the environment. There are many well-known examples in application areas such as medical devices, aircraft flight control, weapons and nuclear systems (Knight, 2003).
Sector	A sector is one of a few general segments in the economy within which a large group of companies can be categorized. An economy can be broken down into about a dozen sectors, which can describe nearly all of the business activity in that economy. Economists can conduct a deeper analysis of the economy by looking at each individual sector. (https://www.investopedia.com/ask/answers/05/industrysector.asp)
Software Development Life Cycle	The software development life cycle (SDLC) is a framework defining tasks performed at each step in the software development process. SDLC is a structure followed by a development team within the software organization. It consists of a detailed plan describing how to develop, maintain and replace specific software. The life cycle defines a methodology for improving the quality of software and the overall development process. The software development life cycle is also known as the software development process. (https://www.techopedia.com/definition/22193/software-development-life-cycle-sdlc)
Traditional development/plan-driven development	A style of development that attempts to plan for and anticipate up front all of the features a user might want in the end product and to determine how best to build those features. The work plan is based on execution of a sequential set of work-specific phases. (https://innolution.com/resources/glossary/plan-driven-process)
Utilitarian	Designed to be useful or practical rather than attractive.
Value	the regard that something is held to deserve; the importance, worth, or usefulness of something.

Appendix B: Questionnaire

Appendix B.1: Finalized questionnaire

The finalized and used questionnaire can be found online and printed at:
https://form.jotformeu.com/91633300544348



Software traceability

Tracing the value of software traceability in software development.

Utrecht University

Welcome to the survey about the value of software traceability in practice.

With this **anonymous** questionnaire, we aim to gather **your view and needs** regarding traceability in the software development process.

This survey is part of a research project that is being conducted at Utrecht University, with the main objective of exploring the value of software traceability based on the perspective of practitioners. With the results, we aim to find **gaps** between **research** and software traceability **practices** with the goal to increase traceability **support for practitioners**.

To this end, we kindly ask you to fill in this questionnaire.

You are eligible to fill in the survey if you are involved in the software development process, be it as a Product Owner, Scrum Master, Developer, Architect, Project manager, Designer, Tester, Requirements Analyst, or any other role closely involved with the development process.

Thank you for your participation!

For questions about the research or issues with filling in the questionnaire, you can send a mail to traceabilitysurvey@gmail.com

Jimmy (Jin Yang) Hu,
Dr. Marcela Ruiz,
Dr. Fabiano Dalpiaz.

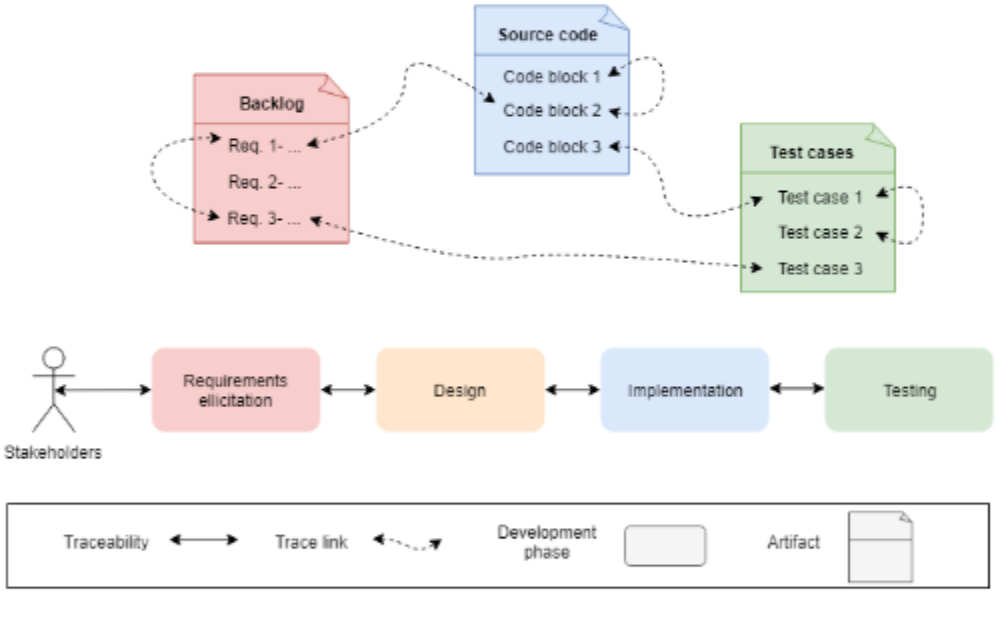
Next

What is software traceability?

In the context of this research, we see software traceability as the ability to **create, maintain, and use relations/links between diverse software development artifacts** such as requirements, source code, test cases, architecture, and any other development artifacts.

These links, for example, enable determining (tracing) the origin of a requirement, or knowing which code fragments relate to which test cases or bug reports. Traceability links can be established and retrieved either manually, or with the use of automated software tools.

An illustration is provided below.



When answering questions in the survey, please provide responses based on the artifacts you are involved with.

Back Next

Demographic questions

In which country do you currently reside? *

Netherlands

What is the size of your company? *

Micro (1-9)

The following questions are about the projects or products that you are generally involved in.

Which role are you typically assigned to during software development projects? (An additional textbox will pop up when choosing 'other'.) *

Product Owner

On average, how many people are involved in the software development projects you participate in? *

10-29 people

Think of the people that interact with software development artifacts such as requirements, architecture, code, test cases, user feedback, user manuals.

In the remainder of the survey please provide answers based on your experience in projects of the size that you have selected.

Which software development paradigm is normally used in the projects you are involved with? *

- (•) Agile: Scrum, Kanban, XP, ...
- ( ) Traditional: Waterfall, Incremental Development, V-model, Spiral, ...
- ( ) Mixed
- ( ) Other

Would you characterize the industry you work in to be highly regulated? (Examples: Healthcare, Automotive, Aerospace, Finance, ...) *

Would you characterize the industry you work in to be highly regulated? (Examples: Healthcare, Automotive, Aerospace, Finance, ...) *

- (•) Yes
- ( ) No

Would you characterize your software as safety-critical? (Examples: software for medical, automotive, railway, aviation, infrastructure, and nuclear systems, ...) *

- ( ) Yes
- (•) No

How often is software traceability performed in the projects that you are involved in? When answering 'Always' or 'Sometimes' one extra question will pop up. *

- (•) Always
- ( ) Sometimes
- ( ) Never

Your reason(s) for performing traceability (multiple answers possible): *

- [x] because of mandate (regulations/ISO)
- [x] on request of customer
- [ ] for the expected benefits
- [ ] Other

Back Next

Current situation

Please indicate **your** level of agreement with the following statements based on your **opinion and experiences** of your **current or most recent projects** that you are typically involved in.

N/A if not applicable or if you do not want to answer.

Traceability Management *

	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	N/A
1. Traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices	○	○	○	○	○
2. There is insufficient guidance within the company on how to establish traceability	○	○	○	○	○
3. Traceability costs outweigh the expected benefits	○	○	○	○	○
4. It is difficult to access and obtain information sources (people and artifacts) to be able to establish traceability	○	○	○	○	○
5. Traceability is mostly performed in an ad-hoc and non managed fashion	○	○	○	○	○

Social collaboration practices *

	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	N/A
6. Traceability is of high importance for the software development process	○	○	○	○	○
7. The allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability	○	○	○	○	○
8. There is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability	○	○	○	○	○
9. It is unclear which roles are responsible for traceability	○	○	○	○	○

Technical aspects *

	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	N/A
10. Traceability tools do not satisfy our traceability needs	○	○	○	○	○
11. It is difficult to establish traceability because development artifacts are stored in multiple locations/repositories	○	○	○	○	○
12. The tools used for traceability are too complex to use effectively and to integrate with our existing tools	○	○	○	○	○
13. Traceability is mostly performed manually	○	○	○	○	○

Back Next

Need of establishing traceability

Please **prioritize** the following statements based on **your needs** of your **current or most recent projects** that you are typically involved in.

The possible prioritizations are:

Not needed -- not of importance
Nice to have -- desirable but not necessary
Should have -- important but not critical
Must have -- critical

Management needs *

	Not needed	Nice to have	Should have	Must have	N/A
1. There is a need to reduce the costs of traceability	○	○	○	○	○
2. There is a need for more guidance in the company regarding traceability	○	○	○	○	○
3. There is a need to have a more clear overview of the costs and benefits regarding traceability	○	○	○	○	○
4. There is a need to have easier access to information sources to be able to establish and maintain traceability	○	○	○	○	○
5. There is a need to perform traceability in a more managed fashion	○	○	○	○	○

Social collaboration needs *

Social collaboration needs *

	Not needed	Nice to have	Should have	Must have	N/A
6. There is a need to increase the awareness of the importance of traceability	○	○	○	○	○
7. There is a need for more staff, time, and rescources to properly establish and maintain traceability	○	○	○	○	○
8. There is a need for more collaboration between involved stakeholder teams regarding traceability	○	○	○	○	○
9. There is a need for more communication about who are responsible for traceability	○	○	○	○	○

Technical needs *

	Not needed	Nice to have	Should have	Must have	N/A
10. There is a need for better traceability tools	○	○	○	○	○
11. There is a need for a more centralized development artifact repository to establish traceability more easily	○	○	○	○	○
12. There is a need for less complex traceability tools and easier integration with our exisiting tools	○	○	○	○	○
13. There is a need for more traceability automation	○	○	○	○	○

Back Next

Comments and feedback

Feel free to leave any comments regarding this questionnaire or the topic of traceability.

When finished, do not forget to press on the **submit button!**

Is there anything else you would like to add?

If you are interested in the results of this research and would like to receive them, please provide an email address.

Would you be interested in helping us further with the research with a short interview about the context and reasons behind your answers? If yes, please provide an email address in the above text field.

- Yes
- No

Would you be interested in helping us further with the research with a short interview about the context and reasons behind your answers? If yes, please provide an email address in the above text field.

- Yes
- No

The contact information will only be used for distributing the results. In case of interest for an interview, it will also be used as a reference to your answers to be able to conduct a more relevant interview. The results of the interview will be anonymized as well.

Submit

Powered by JotForm

Back

Appendix C: Interviews

Appendix C.1: validation interview informed consent

Dear participant,

You have been asked to participate in the validation of the questionnaire about the value of software traceability in practice. The results are used to improve the content, design, and user experience of the questionnaire. This interview and validation is part of a research project that is performed by J.Y. Hu from the Master Business Informatics at the University of Utrecht. The goal of the research project is to explore the value of software traceability based on the current situation and needs of practitioners so that possible gaps between research and support for practitioners can be found and improved based on what is needed.

This validation is being recorded to help with the note-taking. The recording is only used by me and my colleagues and is not made publicly available. In addition:

- All the information/recordings are held confidential and will be anonymized.
- Your participation is voluntary and you can stop at any time.
- If you don't want to answer a question for any reason, you can always skip it.

I have read this form and agree,

…… …….

Date Signature participant

Appendix C.2: validation interview protocol

Validation and semi-structured interview Protocol
Semi structured interview by Jimmy (Jin Yang) Hu

Note: All these items listed below are guidelines, which means that some of the questions can be asked differently depending on the interviewee's reaction and information he or she has already provided. In addition, the interviewees were already notified that this is an interview to validate a questionnaire and that there is a short think-aloud session during the meeting as well.

BEFORE INTERVIEW**:**

- Ask for permission to record this interview.
    - This interview is being recorded to help with note-taking. The recording is only used by me and my colleagues of this project, and is not made publicly available. In addition we ask you to sign an informed consent which states:
        - All the information/recordings are held confidential and will be anonymized.
        - You participation is voluntary and you can stop at any time
        - If you don't want to answer a question for any reason, you can always skip it.

- Date of interview:

- Time of interview:

START INTERVIEW:

- Introduction of myself (Jimmy)

Participant background

Participant name:

- Can you tell me something about yourself?
    - Can you briefly describe your work experience?
    - Education background?
    - How long have you worked for this organization?
    - Can you briefly describe your role in the organization?
    - Depending on the answer: Can you briefly describe your current or past Involvement with the software development process?
    - How did you get involved with the current roles?

- Explain and describe think-aloud, setup, and let the interviewee fill in the survey.
    - Interviewee fills in the survey as how they would normally fill it in, while thinking out loud. Interviewer will not be involved during this part, unless there are technical issues/other issues.
    - Interviewee also submits the results

- Take think aloud session notes (Jimmy)
    - Start time:
    - End time:
    - Rest in the notebook

Retrospective questions

- Thank interviewee for performing the think-aloud session.

- As you most likely have noticed the questionnaire is about software traceability. Based on this session, I have several retrospective questions about the questionnaire, your questionnaire answers and context:

Depending on how the think aloud session went, some of these questions will be asked, asked differently or not asked at all. The interviewee can click through the questionnaire again while answering some of the questions below. Depending on their answers of their filled in the questionnaire, or their answers on the questions below, the interviewer will probe for further explanation about the reasoning and the context of the interviewee where necessary or applicable.

- Was the questionnaire easy or difficult to fill in, and why?

- Have you dealt with traceability before during your work?
    - What is your experience with traceability in the software development in general?

Questionnaire
Content:

- Were the descriptions and explanations clear? If not, can you briefly explain which of the descriptions and why?

- Were the questions and statements worded clearly?
- Did the questions and statements make sense to you?
- Did you have trouble answering a certain question, and can you briefly explain why?
- Did the possible answer/categories make sense to you? If not, which ones and can you briefly explain why?
- Did you notice any ambiguities regarding the concepts used? Were you familiar with the concepts, or did some of the concepts confuse you?
- Did you feel like there was any overlap between certain questions?
- Did you notice any spelling errors?
- Would you fill in this questionnaire based on the topic and introduction and why?
- Do you think Practitioners in the software development/same field will be able to fill in the questionnaire?

Design:

- Is the survey structured in an intuitive way and did you have any trouble navigating through the questionnaire?
- Were there parts of the survey that struck you as odd? If so, which parts and why?
- Is the overall layout pleasant to the eyes?

Additional comments/improvements

- Do you have any tips to improve the questionnaire?
- Any comments regarding traceability?
- Thank interviewee for helping with the validation and don't forget the token for appreciation.

Appendix C.3: survey interview informed consent

Informed consent Value of Traceability

Dear participant,

You have been asked to participate in an interview about the value of software traceability in practice. This interview is part of a research project that is performed by J.Y. Hu from the Master Business Informatics at the University of Utrecht. The goal of the research project is to explore the value of software traceability based on the current situation and needs of practitioners so that possible gaps between research and support for practitioners can be found and improved based on what is needed.

The results of the interview will be anonymized and used to supplement the survey about software traceability which you have also filled in previously.

This interview is being recorded to help with the note-taking. The recording is only used by me and my colleagues and is not made publicly available. In addition:

- All the information/recordings are held confidential and will be anonymized.
- Your participation is voluntary and you can stop at any time.
- If you don't want to answer a question for any reason, you can always skip it.

I have read this form and agree,

…… ……..

Date … Signature participant

Appendix C.4: survey interview protocol

Semi-structured interview Protocol
Semi structured interview by Jimmy (Jin Yang) Hu

Note: All these items listed below are guidelines, which means that some of the questions can be asked differently depending on the interviewee's reaction and information he or she has already provided. In addition, the interviewees were already notified that this is an interview about the context and rationale behind their answers on the traceability questionnaire.

BEFORE INTERVIEW**:**

- Ask for permission to record this interview.
    - This interview is being recorded to help with note-taking. The recording is only used by me and my colleagues of this project, and is not made publicly available. In addition we ask you to sign an informed consent which states:
        - All the information/recordings are held confidential and will be anonymized.
        - You participation is voluntary and you can stop at any time
        - If you don't want to answer a question for any reason, you can always skip it.

- Date of interview:

- Time of interview:

- Participant name:

START INTERVIEW:

- Introduction of interviewer (Jimmy) (~5 minutes)
    - Important to make clear that we would like to cover every important topic as indicated in the slide. In case the interviewee would like to discuss a certain topic in even more detail, that a follow up meeting can be arranged.

Questions in bold are the overarching open questions, the sub questions are mainly used to 'steer' or keep the conversation on track. Not every subquestion will be asked, it depends on what has been described already by the interviewee and if it is applicable or not.

Initial background and project characteristics (10-15 minutes)

- **Can you tell me something about yourself?**
  - Can you briefly describe your work experience? In years and roles
  - Education background? Titles

- **Can you briefly tell me about your company?**
  - Can you briefly tell me about the sector/industry your company is in and which main types of services/products are offered by the company? (Mobile, software, etc.)
  - How long have you worked for this organization/project?
  - Can you briefly tell me about the company structure and culture? Flat, hierarchical

- **Were/are you familiar with the concept of software traceability?**
  - Which artifacts do you mainly think of when talking about traceability?
  - Which artifacts would you be responsible for in the typical projects you are involved with?
  - Which artifacts are the most important to establish traceability between for you and why?

- **In the survey you indicated that the typical software developed is safety-critical, or not safety critical, can you provide a brief example what the software is used for and in what way it is safety-critical or not safety-critical? (Could be critical for the life of users, or critical because of sensitive data, or high risks in other kinds of potential losses)**

- **In the survey you indicated that traceability is Always/sometimes/never performed,**
  - In case of Always/Sometimes, you indicated that it is performed for the Benefits/because of mandate/on request of customer, can you briefly describe a common scenario or which benefits are expected?

- **In the survey you mentioned that the typical software development paradigm used is Agile/Traditional/Mixed, could you describe which development method/process is followed (scrum, XP, kanban, waterfall, ….), and briefly describe the process from when a wish/requirement comes in till its deployment? (to get a general picture of the process and information flow between the teams)**
  - Iterations? Self organizing teams?
  - Local or geographically dispersed or outsourced teams?

Management practices (~10 minutes)

Can you tell me how the management statements in the survey relate to the current management practices in your projects and:

1. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that traceability costs in money, time, and effort are the main inhibitor for adopting (more mature) traceability practices

a. Which costs between time, effort and money is most problematic?

And why it is [not needed/nice to have/ should have/ must have] to reduce the costs?

b. Which costs have to be reduced?

2. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that there is insufficient guidance within the company to establish traceability
   And why it is [not needed/nice to have/ should have/ must have] to increase the guidance in the company?

3. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that traceability costs outweigh the expected benefits
   a. Which benefits do you expect to be gained from traceability practices?

   And why it is [not needed/nice to have/ should have/ must have] to have a more clear overview of the costs and benefits regarding traceability?

4. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that it is difficult to access and obtain information sources to establish traceability
   And why it is [not needed/nice to have/ should have/ must have] to have easier access to information sources to be able to establish and maintain traceability?

5. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that Traceability is mostly performed in an ad-hoc and non managed fashion,
   And why it is [not needed/nice to have/ should have/ must have] to perform traceability in a more managed fashion?

Social collaboration practices (~10 minutes)

Can you tell me how the social collaboration statements in the survey relate to the current collaboration practices in your projects and why:

1. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that traceability is of high importance for the software development process,
   And why it is [not needed/nice to have/ should have/ must have] to increase the awareness of the importance of traceability?

2. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that the allocation of time, staff and resources are often insufficient to be able to properly establish and maintain traceability,
   And why it is [not needed/nice to have/ should have/ must have] for more staff, time, and resources to properly establish and maintain traceability?
   a. Which of the three, time, staff, and resources, are sufficient and insufficient?

3. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that there is a lack of collaboration between involved stakeholder teams in regards to establishing and maintaining traceability,
   And why it is [not needed/nice to have/ should have/ must have] for more collaboration between involved stakeholder teams?

4. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that it is unclear which roles are responsible for traceability

And why it is [not needed/nice to have/ should have/ must have] to have more communication about who are responsible for traceability?

Technology (~10 minutes)

Can you tell me how the technology statements in the survey relate to the current technology practices and tools used in your projects and why:

1. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that Traceability tools do not satisfy your needs,
   And why it is [not needed/nice to have/ should have/ must have] for better traceability tools?
   a. Which tools are being used for traceability purposes?

2. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that it is difficult to establish traceability because development artifacts are stored in multiple locations/repositories,
   And why it is [not needed/nice to have/ should have/ must have] to have a more centralized development artifacts repository to establish traceability more easily?

3. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that the tools currently used for traceability are too complex to use effectively and to integrate with existing tools,
   And why it is [not needed/nice to have/ should have/ must have] to make them less complex and easier to integrate with your existing tools?

4. Why you [strongly disagree/ somewhat disagree/ somewhat agree/ strongly agree] with that Traceability is mostly performed manually,
   And why it is [not needed/nice to have/ should have/ must have] for more traceability automation?

Importance of traceability in the SDLC (~10 minutes)

- In which software development stages would you say that traceability is the most important/critical? (Show slide with the SDLC phases, ask if there are any phases missing)
- Or between which of the phases?
- And why?

Additional comments/improvements

- Do you have any additional comments?

- Thank interviewee and don't forget the token for appreciation.

Appendix D: Total sample: descriptive summary tables

D.1 Overall current statements summary statistics

D.1.1 Perceived current situation: statement frequency table with NA

Statement	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	<NA>	Total
Current_1	2	9	23	20	1	55
Current_2	3	14	25	7	6	55
Current_3	13	20	15	5	2	55
Current_4	7	26	15	6	1	55
Current_5	12	9	19	11	4	55
Current_6	3	5	23	24	0	55
Current_7	6	10	21	14	4	55
Current_8	6	17	21	8	3	55
Current_9	6	14	17	13	5	55
Current_10	6	16	19	4	10	55
Current_11	8	18	15	10	4	55
Current_12	7	20	14	6	8	55
Current_13	4	9	19	20	3	55

D.1.2 Perceived current situation: statement percentile table with NA

Statement	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	<NA>	Total
Current_1	3.64%	16.36%	41.82%	36.36%	1.82%	100.00%
Current_2	5.45%	25.45%	45.45%	12.73%	10.91%	100.00%
Current_3	23.64%	36.36%	27.27%	9.09%	3.64%	100.00%
Current_4	12.73%	47.27%	27.27%	10.91%	1.82%	100.00%
Current_5	21.82%	16.36%	34.55%	20.00%	7.27%	100.00%
Current_6	5.45%	9.09%	41.82%	43.64%	0.00%	100.00%
Current_7	10.91%	18.18%	38.18%	25.45%	7.27%	100.00%
Current_8	10.91%	30.91%	38.18%	14.55%	5.45%	100.00%
Current_9	10.91%	25.45%	30.91%	23.64%	9.09%	100.00%
Current_10	10.91%	29.09%	34.55%	7.27%	18.18%	100.00%
Current_11	14.55%	32.73%	27.27%	18.18%	7.27%	100.00%
Current_12	12.73%	36.36%	25.45%	10.91%	14.55%	100.00%
Current_13	7.27%	16.36%	34.55%	36.36%	5.45%	100.00%

D.1.3 Perceived current situation: statement frequency table without NA

Statement	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	Total
Current_1	2	9	23	20	54

Current_2	3	14	25	7	49
Current_3	13	20	15	5	53
Current_4	7	26	15	6	54
Current_5	12	9	19	11	51
Current_6	3	5	23	24	55
Current_7	6	10	21	14	51
Current_8	6	17	21	8	52
Current_9	6	14	17	13	50
Current_10	6	16	19	4	45
Current_11	8	18	15	10	51
Current_12	7	20	14	6	47
Current_13	4	9	19	20	52

D.1.4 Perceived current situation: statement percentile table without NA

Statement	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	Total
Current_1	3.70%	16.67%	42.59%	37.04%	100.00%
Current_2	6.12%	28.57%	51.02%	14.29%	100.00%
Current_3	24.53%	37.74%	28.30%	9.43%	100.00%
Current_4	12.96%	48.15%	27.78%	11.11%	100.00%
Current_5	23.53%	17.65%	37.25%	21.57%	100.00%
Current_6	5.45%	9.09%	41.82%	43.64%	100.00%
Current_7	11.76%	19.61%	41.18%	27.45%	100.00%
Current_8	11.54%	32.69%	40.38%	15.38%	100.00%
Current_9	12.00%	28.00%	34.00%	26.00%	100.00%
Current_10	13.33%	35.56%	42.22%	8.89%	100.00%
Current_11	15.69%	35.29%	29.41%	19.61%	100.00%
Current_12	14.89%	42.55%	29.79%	12.77%	100.00%
Current_13	7.69%	17.31%	36.54%	38.46%	100.00%

D.1.5 Perceived current situation: Mean summary table

statement	mean
Current_1	3.13
Current_2	2.73
Current_3	2.23
Current_4	2.37
Current_5	2.57
Current_6	3.24
Current_7	2.84
Current_8	2.60
Current_9	2.74
Current_10	2.47

Current_11	2.53
Current_12	2.40
Current_13	3.06

D.2 Overall need statements summary statistics

D.2.1 Perceived needs: statement frequency table with NA

statement	Not needed	Nice to have	Should have	Must have	<NA>
Need_1	11	18	15	9	2
Need_2	9	15	14	14	3
Need_3	2	15	24	12	2
Need_4	9	13	19	12	2
Need_5	6	13	16	19	1
Need_6	7	13	16	19	0
Need_7	14	16	16	7	2
Need_8	6	16	20	10	3
Need_9	8	11	20	13	3
Need_10	11	16	13	13	2
Need_11	15	18	12	10	0
Need_12	7	18	18	10	2
Need_13	5	18	14	18	0

D.2.2 Perceived needs: statement percentile table with NA

statement	Not needed	Nice to have	Should have	Must have	<NA>
Need_1	20.00%	32.73%	27.27%	16.36%	3.64%
Need_2	16.36%	27.27%	25.45%	25.45%	5.45%
Need_3	3.64%	27.27%	43.64%	21.82%	3.64%
Need_4	16.36%	23.64%	34.55%	21.82%	3.64%
Need_5	10.91%	23.64%	29.09%	34.55%	1.82%
Need_6	12.73%	23.64%	29.09%	34.55%	0.00%
Need_7	25.45%	29.09%	29.09%	12.73%	3.64%
Need_8	10.91%	29.09%	36.36%	18.18%	5.45%
Need_9	14.55%	20.00%	36.36%	23.64%	5.45%
Need_10	20.00%	29.09%	23.64%	23.64%	3.64%
Need_11	27.27%	32.73%	21.82%	18.18%	0.00%
Need_12	12.73%	32.73%	32.73%	18.18%	3.64%
Need_13	9.09%	32.73%	25.45%	32.73%	0.00%

D.2.3 Perceived needs: statement frequency table without NA

statement	Not needed	Nice to have	Should have	Must have	Total
Need_1	11	18	15	9	53
Need_2	9	15	14	14	52
Need_3	2	15	24	12	53
Need_4	9	13	19	12	53
Need_5	6	13	16	19	54
Need_6	7	13	16	19	55
Need_7	14	16	16	7	53
Need_8	6	16	20	10	52
Need_9	8	11	20	13	52
Need_10	11	16	13	13	53
Need_11	15	18	12	10	55
Need_12	7	18	18	10	53
Need_13	5	18	14	18	55

D.2.4 Perceived needs: statement percentile table without NA

statement	Not needed	Nice to have	Should have	Must have	Total
Need_1	20.75%	33.96%	28.30%	16.98%	100.00%
Need_2	17.31%	28.85%	26.92%	26.92%	100.00%
Need_3	3.77%	28.30%	45.28%	22.64%	100.00%
Need_4	16.98%	24.53%	35.85%	22.64%	100.00%
Need_5	11.11%	24.07%	29.63%	35.19%	100.00%
Need_6	12.73%	23.64%	29.09%	34.55%	100.00%
Need_7	26.42%	30.19%	30.19%	13.21%	100.00%
Need_8	11.54%	30.77%	38.46%	19.23%	100.00%
Need_9	15.38%	21.15%	38.46%	25.00%	100.00%
Need_10	20.75%	30.19%	24.53%	24.53%	100.00%
Need_11	27.27%	32.73%	21.82%	18.18%	100.00%
Need_12	13.21%	33.96%	33.96%	18.87%	100.00%
Need_13	9.09%	32.73%	25.45%	32.73%	100.00%

D.2.5 Perceived needs: Mean summary table

statement	mean
Need_1	2.42
Need_2	2.63
Need_3	2.87
Need_4	2.64
Need_5	2.89
Need_6	2.85

Need_7	2.30
Need_8	2.65
Need_9	2.73
Need_10	2.53
Need_11	2.31
Need_12	2.58
Need_13	2.82

Appendix E: Agile vs non-Agile: descriptive summary tables

E.1 Current situation summary tables

E.1.1 Frequency summary table with NA

statement	Paradigm	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	<NA>	Total
Current_1	Agile	2	4	16	14	1	37
Current_1	Mixed	0	5	5	5	0	15
Current_1	Traditional	0	0	2	1	0	3
Current_2	Agile	1	10	18	4	4	37
Current_2	Mixed	1	3	6	3	2	15
Current_2	Traditional	1	1	1	0	0	3
Current_3	Agile	8	12	11	4	2	37
Current_3	Mixed	4	7	3	1	0	15
Current_3	Traditional	1	1	1	0	0	3
Current_4	Agile	5	20	9	2	1	37
Current_4	Mixed	1	6	6	2	0	15
Current_4	Traditional	1	0	0	2	0	3
Current_5	Agile	7	7	13	8	2	37
Current_5	Mixed	3	2	6	3	1	15
Current_5	Traditional	2	0	0	0	1	3
Current_6	Agile	3	5	16	13	0	37
Current_6	Mixed	0	0	7	8	0	15
Current_6	Traditional	0	0	0	3	0	3
Current_7	Agile	4	9	12	9	3	37
Current_7	Mixed	2	1	7	4	1	15
Current_7	Traditional	0	0	2	1	0	3
Current_8	Agile	5	10	14	6	2	37
Current_8	Mixed	1	6	6	1	1	15
Current_8	Traditional	0	1	1	1	0	3
Current_9	Agile	2	11	11	10	3	37
Current_9	Mixed	3	3	4	3	2	15
Current_9	Traditional	1	0	2	0	0	3

Current_10	Agile	4	11	11	3	8	37
Current_10	Mixed	2	4	6	1	2	15
Current_10	Traditional	0	1	2	0	0	3
Current_11	Agile	6	11	10	6	4	37
Current_11	Mixed	2	6	3	4	0	15
Current_11	Traditional	0	1	2	0	0	3
Current_12	Agile	5	12	10	5	5	37
Current_12	Mixed	1	7	3	1	3	15
Current_12	Traditional	1	1	1	0	0	3
Current_13	Agile	3	4	16	12	2	37
Current_13	Mixed	1	5	3	5	1	15
Current_13	Traditional	0	0	0	3	0	3

E.1.2 Percentage summary table with NA

statement	Paradigm	total_n	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	<NA>
Current_1	Agile	37	5.41%	10.81%	43.24%	37.84%	2.70%
Current_1	Mixed	15	0.00%	33.33%	33.33%	33.33%	0.00%
Current_1	Traditional	3	0.00%	0.00%	66.67%	33.33%	0.00%
Current_2	Agile	37	2.70%	27.03%	48.65%	10.81%	10.81%
Current_2	Mixed	15	6.67%	20.00%	40.00%	20.00%	13.33%
Current_2	Traditional	3	33.33%	33.33%	33.33%	0.00%	0.00%
Current_3	Agile	37	21.62%	32.43%	29.73%	10.81%	5.41%
Current_3	Mixed	15	26.67%	46.67%	20.00%	6.67%	0.00%
Current_3	Traditional	3	33.33%	33.33%	33.33%	0.00%	0.00%
Current_4	Agile	37	13.51%	54.05%	24.32%	5.41%	2.70%
Current_4	Mixed	15	6.67%	40.00%	40.00%	13.33%	0.00%
Current_4	Traditional	3	33.33%	0.00%	0.00%	66.67%	0.00%
Current_5	Agile	37	18.92%	18.92%	35.14%	21.62%	5.41%
Current_5	Mixed	15	20.00%	13.33%	40.00%	20.00%	6.67%
Current_5	Traditional	3	66.67%	0.00%	0.00%	0.00%	33.33%
Current_6	Agile	37	8.11%	13.51%	43.24%	35.14%	0.00%
Current_6	Mixed	15	0.00%	0.00%	46.67%	53.33%	0.00%
Current_6	Traditional	3	0.00%	0.00%	0.00%	100.00%	0.00%
Current_7	Agile	37	10.81%	24.32%	32.43%	24.32%	8.11%
Current_7	Mixed	15	13.33%	6.67%	46.67%	26.67%	6.67%
Current_7	Traditional	3	0.00%	0.00%	66.67%	33.33%	0.00%
Current_8	Agile	37	13.51%	27.03%	37.84%	16.22%	5.41%
Current_8	Mixed	15	6.67%	40.00%	40.00%	6.67%	6.67%
Current_8	Traditional	3	0.00%	33.33%	33.33%	33.33%	0.00%
Current_9	Agile	37	5.41%	29.73%	29.73%	27.03%	8.11%
Current_9	Mixed	15	20.00%	20.00%	26.67%	20.00%	13.33%
Current_9	Traditional	3	33.33%	0.00%	66.67%	0.00%	0.00%

Current_10	Agile	37	10.81%	29.73%	29.73%	8.11%	21.62%
Current_10	Mixed	15	13.33%	26.67%	40.00%	6.67%	13.33%
Current_10	Traditional	3	0.00%	33.33%	66.67%	0.00%	0.00%
Current_11	Agile	37	16.22%	29.73%	27.03%	16.22%	10.81%
Current_11	Mixed	15	13.33%	40.00%	20.00%	26.67%	0.00%
Current_11	Traditional	3	0.00%	33.33%	66.67%	0.00%	0.00%
Current_12	Agile	37	13.51%	32.43%	27.03%	13.51%	13.51%
Current_12	Mixed	15	6.67%	46.67%	20.00%	6.67%	20.00%
Current_12	Traditional	3	33.33%	33.33%	33.33%	0.00%	0.00%
Current_13	Agile	37	8.11%	10.81%	43.24%	32.43%	5.41%
Current_13	Mixed	15	6.67%	33.33%	20.00%	33.33%	6.67%
Current_13	Traditional	3	0.00%	0.00%	0.00%	100.00%	0.00%

E.1.3 Frequency summary table without NA

statement	Paradigm	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	Total
Current_1	Agile	2	4	16	14	36
Current_1	Mixed	0	5	5	5	15
Current_1	Traditional	0	0	2	1	3
Current_2	Agile	1	10	18	4	33
Current_2	Mixed	1	3	6	3	13
Current_2	Traditional	1	1	1	0	3
Current_3	Agile	8	12	11	4	35
Current_3	Mixed	4	7	3	1	15
Current_3	Traditional	1	1	1	0	3
Current_4	Agile	5	20	9	2	36
Current_4	Mixed	1	6	6	2	15
Current_4	Traditional	1	0	0	2	3
Current_5	Agile	7	7	13	8	35
Current_5	Mixed	3	2	6	3	14
Current_5	Traditional	2	0	0	0	2
Current_6	Agile	3	5	16	13	37
Current_6	Mixed	0	0	7	8	15
Current_6	Traditional	0	0	0	3	3
Current_7	Agile	4	9	12	9	34
Current_7	Mixed	2	1	7	4	14
Current_7	Traditional	0	0	2	1	3
Current_8	Agile	5	10	14	6	35
Current_8	Mixed	1	6	6	1	14
Current_8	Traditional	0	1	1	1	3
Current_9	Agile	2	11	11	10	34
Current_9	Mixed	3	3	4	3	13
Current_9	Traditional	1	0	2	0	3

Current_10	Agile	4	11	11	3	29
Current_10	Mixed	2	4	6	1	13
Current_10	Traditional	0	1	2	0	3
Current_11	Agile	6	11	10	6	33
Current_11	Mixed	2	6	3	4	15
Current_11	Traditional	0	1	2	0	3
Current_12	Agile	5	12	10	5	32
Current_12	Mixed	1	7	3	1	12
Current_12	Traditional	1	1	1	0	3
Current_13	Agile	3	4	16	12	35
Current_13	Mixed	1	5	3	5	14
Current_13	Traditional	0	0	0	3	3

E.1.4 Percentage summary table without NA

statement	Paradigm	total_n	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree
Current_1	Agile	36	5.56%	11.11%	44.44%	38.89%
Current_1	Mixed	15	0.00%	33.33%	33.33%	33.33%
Current_1	Traditional	3	0.00%	0.00%	66.67%	33.33%
Current_2	Agile	33	3.03%	30.30%	54.55%	12.12%
Current_2	Mixed	13	7.69%	23.08%	46.15%	23.08%
Current_2	Traditional	3	33.33%	33.33%	33.33%	0.00%
Current_3	Agile	35	22.86%	34.29%	31.43%	11.43%
Current_3	Mixed	15	26.67%	46.67%	20.00%	6.67%
Current_3	Traditional	3	33.33%	33.33%	33.33%	0.00%
Current_4	Agile	36	13.89%	55.56%	25.00%	5.56%
Current_4	Mixed	15	6.67%	40.00%	40.00%	13.33%
Current_4	Traditional	3	33.33%	0.00%	0.00%	66.67%
Current_5	Agile	35	20.00%	20.00%	37.14%	22.86%
Current_5	Mixed	14	21.43%	14.29%	42.86%	21.43%
Current_5	Traditional	2	100.00%	0.00%	0.00%	0.00%
Current_6	Agile	37	8.11%	13.51%	43.24%	35.14%
Current_6	Mixed	15	0.00%	0.00%	46.67%	53.33%
Current_6	Traditional	3	0.00%	0.00%	0.00%	100.00%
Current_7	Agile	34	11.76%	26.47%	35.29%	26.47%
Current_7	Mixed	14	14.29%	7.14%	50.00%	28.57%
Current_7	Traditional	3	0.00%	0.00%	66.67%	33.33%
Current_8	Agile	35	14.29%	28.57%	40.00%	17.14%
Current_8	Mixed	14	7.14%	42.86%	42.86%	7.14%
Current_8	Traditional	3	0.00%	33.33%	33.33%	33.33%
Current_9	Agile	34	5.88%	32.35%	32.35%	29.41%
Current_9	Mixed	13	23.08%	23.08%	30.77%	23.08%
Current_9	Traditional	3	33.33%	0.00%	66.67%	0.00%

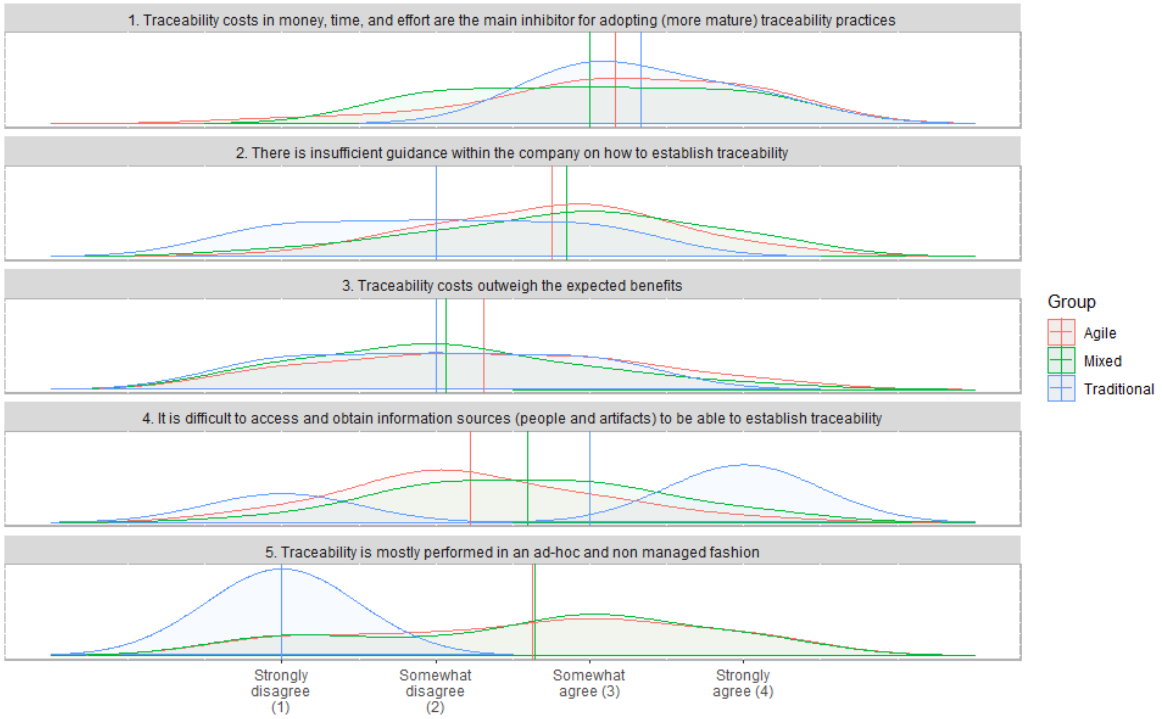
Current_10	Agile	29	13.79%	37.93%	37.93%	10.34%
Current_10	Mixed	13	15.38%	30.77%	46.15%	7.69%
Current_10	Traditional	3	0.00%	33.33%	66.67%	0.00%
Current_11	Agile	33	18.18%	33.33%	30.30%	18.18%
Current_11	Mixed	15	13.33%	40.00%	20.00%	26.67%
Current_11	Traditional	3	0.00%	33.33%	66.67%	0.00%
Current_12	Agile	32	15.63%	37.50%	31.25%	15.63%
Current_12	Mixed	12	8.33%	58.33%	25.00%	8.33%
Current_12	Traditional	3	33.33%	33.33%	33.33%	0.00%
Current_13	Agile	35	8.57%	11.43%	45.71%	34.29%
Current_13	Mixed	14	7.14%	35.71%	21.43%	35.71%
Current_13	Traditional	3	0.00%	0.00%	0.00%	100.00%

E.1.5 Mean summary table current situation

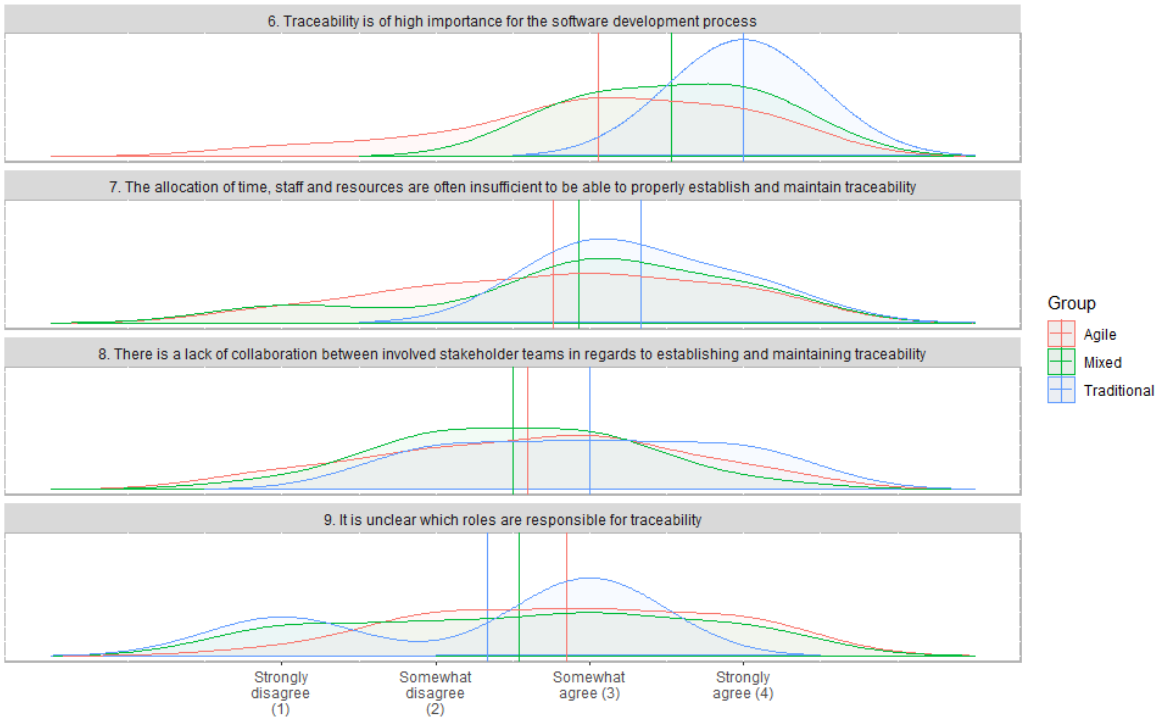
Statement	Agile	Mixed	Traditional	Agile_Mixed	Agile_Traditional	Mixed_Traditional
Current_1	3.17	3.00	3.33	0.17	-0.17	-0.33
Current_2	2.76	2.85	2.00	-0.09	0.76	0.85
Current_3	2.31	2.07	2.00	0.25	0.31	0.07
Current_4	2.22	2.60	3.00	-0.38	-0.78	-0.40
Current_5	2.63	2.64	1.00	-0.01	1.63	1.64
Current_6	3.05	3.53	4.00	-0.48	-0.95	-0.47
Current_7	2.76	2.93	3.33	-0.16	-0.57	-0.40
Current_8	2.60	2.50	3.00	0.10	-0.40	-0.50
Current_9	2.85	2.54	2.33	0.31	0.52	0.21
Current_10	2.45	2.46	2.67	-0.01	-0.22	-0.21
Current_11	2.48	2.60	2.67	-0.12	-0.18	-0.07
Current_12	2.47	2.33	2.00	0.14	0.47	0.33
Current_13	3.06	2.86	4.00	0.20	-0.94	-1.14

E.1.6 Density graphs

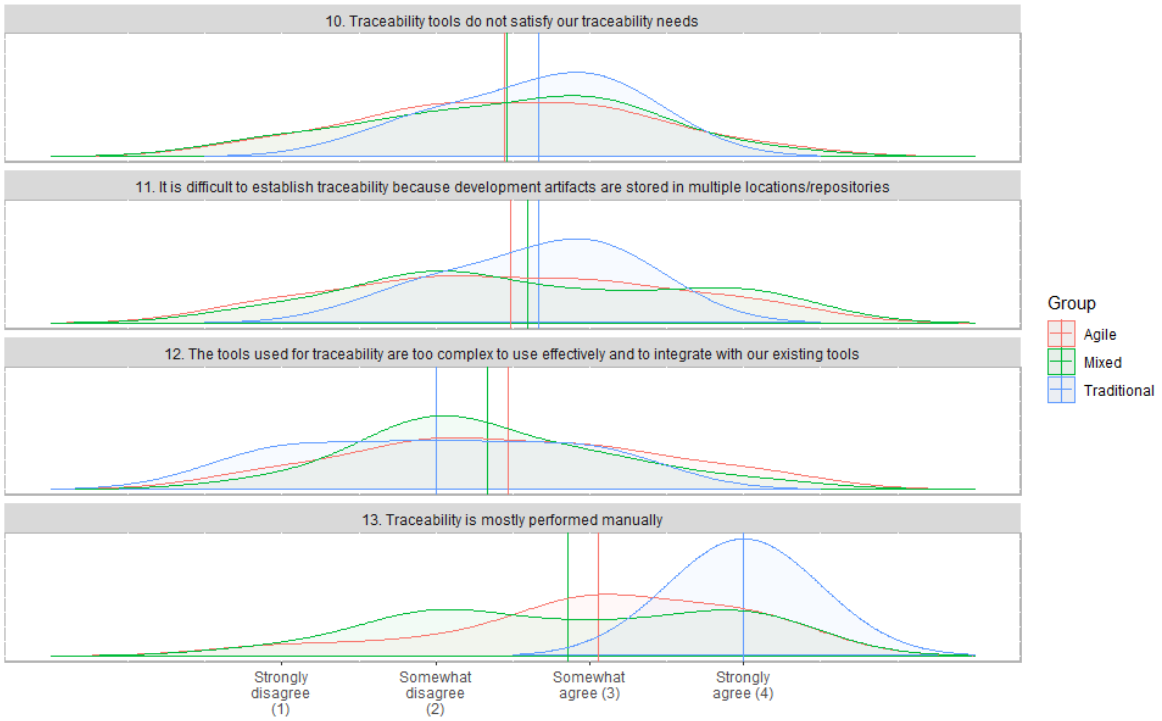
Management



Social



Technical



E.2 Need summary tables

E.2.1 Frequency summary table with NA

statement	Paradigm	Not needed	Nice to have	Should have	Must have	<NA>	Total
Need_1	Agile	8	10	12	6	1	37
Need_1	Mixed	2	7	2	3	1	15
Need_1	Traditional	1	1	1	0	0	3
Need_2	Agile	7	13	7	9	1	37
Need_2	Mixed	1	2	7	3	2	15
Need_2	Traditional	1	0	0	2	0	3
Need_3	Agile	2	11	13	9	2	37
Need_3	Mixed	0	3	9	3	0	15
Need_3	Traditional	0	1	2	0	0	3
Need_4	Agile	6	6	15	9	1	37
Need_4	Mixed	3	5	4	2	1	15
Need_4	Traditional	0	2	0	1	0	3
Need_5	Agile	4	10	8	14	1	37
Need_5	Mixed	2	3	7	3	0	15
Need_5	Traditional	0	0	1	2	0	3
Need_6	Agile	5	11	9	12	0	37

Need_6	Mixed	2	2	5	6	0	15
Need_6	Traditional	0	0	2	1	0	3
Need_7	Agile	11	11	11	3	1	37
Need_7	Mixed	3	4	4	3	1	15
Need_7	Traditional	0	1	1	1	0	3
Need_8	Agile	3	12	14	6	2	37
Need_8	Mixed	3	3	6	2	1	15
Need_8	Traditional	0	1	0	2	0	3
Need_9	Agile	5	8	12	10	2	37
Need_9	Mixed	2	3	8	1	1	15
Need_9	Traditional	1	0	0	2	0	3
Need_10	Agile	8	8	10	9	2	37
Need_10	Mixed	3	7	1	4	0	15
Need_10	Traditional	0	1	2	0	0	3
Need_11	Agile	11	13	7	6	0	37
Need_11	Mixed	4	4	4	3	0	15
Need_11	Traditional	0	1	1	1	0	3
Need_12	Agile	4	12	11	8	2	37
Need_12	Mixed	3	6	5	1	0	15
Need_12	Traditional	0	0	2	1	0	3
Need_13	Agile	3	13	10	11	0	37
Need_13	Mixed	2	4	3	6	0	15
Need_13	Traditional	0	1	1	1	0	3

E.2.2 Percentage summary table with NA

statement	Paradigm	total_n	Not needed	Nice to have	Should have	Must have	<NA>
Need_1	Agile	37	21.62%	27.03%	32.43%	16.22%	2.70%
Need_1	Mixed	15	13.33%	46.67%	13.33%	20.00%	6.67%
Need_1	Traditional	3	33.33%	33.33%	33.33%	0.00%	0.00%
Need_2	Agile	37	18.92%	35.14%	18.92%	24.32%	2.70%
Need_2	Mixed	15	6.67%	13.33%	46.67%	20.00%	13.33%
Need_2	Traditional	3	33.33%	0.00%	0.00%	66.67%	0.00%
Need_3	Agile	37	5.41%	29.73%	35.14%	24.32%	5.41%
Need_3	Mixed	15	0.00%	20.00%	60.00%	20.00%	0.00%
Need_3	Traditional	3	0.00%	33.33%	66.67%	0.00%	0.00%
Need_4	Agile	37	16.22%	16.22%	40.54%	24.32%	2.70%
Need_4	Mixed	15	20.00%	33.33%	26.67%	13.33%	6.67%
Need_4	Traditional	3	0.00%	66.67%	0.00%	33.33%	0.00%
Need_5	Agile	37	10.81%	27.03%	21.62%	37.84%	2.70%
Need_5	Mixed	15	13.33%	20.00%	46.67%	20.00%	0.00%
Need_5	Traditional	3	0.00%	0.00%	33.33%	66.67%	0.00%
Need_6	Agile	37	13.51%	29.73%	24.32%	32.43%	0.00%

Need_6	Mixed	15	13.33%	13.33%	33.33%	40.00%	0.00%
Need_6	Traditional	3	0.00%	0.00%	66.67%	33.33%	0.00%
Need_7	Agile	37	29.73%	29.73%	29.73%	8.11%	2.70%
Need_7	Mixed	15	20.00%	26.67%	26.67%	20.00%	6.67%
Need_7	Traditional	3	0.00%	33.33%	33.33%	33.33%	0.00%
Need_8	Agile	37	8.11%	32.43%	37.84%	16.22%	5.41%
Need_8	Mixed	15	20.00%	20.00%	40.00%	13.33%	6.67%
Need_8	Traditional	3	0.00%	33.33%	0.00%	66.67%	0.00%
Need_9	Agile	37	13.51%	21.62%	32.43%	27.03%	5.41%
Need_9	Mixed	15	13.33%	20.00%	53.33%	6.67%	6.67%
Need_9	Traditional	3	33.33%	0.00%	0.00%	66.67%	0.00%
Need_10	Agile	37	21.62%	21.62%	27.03%	24.32%	5.41%
Need_10	Mixed	15	20.00%	46.67%	6.67%	26.67%	0.00%
Need_10	Traditional	3	0.00%	33.33%	66.67%	0.00%	0.00%
Need_11	Agile	37	29.73%	35.14%	18.92%	16.22%	0.00%
Need_11	Mixed	15	26.67%	26.67%	26.67%	20.00%	0.00%
Need_11	Traditional	3	0.00%	33.33%	33.33%	33.33%	0.00%
Need_12	Agile	37	10.81%	32.43%	29.73%	21.62%	5.41%
Need_12	Mixed	15	20.00%	40.00%	33.33%	6.67%	0.00%
Need_12	Traditional	3	0.00%	0.00%	66.67%	33.33%	0.00%
Need_13	Agile	37	8.11%	35.14%	27.03%	29.73%	0.00%
Need_13	Mixed	15	13.33%	26.67%	20.00%	40.00%	0.00%
Need_13	Traditional	3	0.00%	33.33%	33.33%	33.33%	0.00%

E.2.3 Frequency summary table without NA

statement	Paradigm	Not needed	Nice to have	Should have	Must have	Total
Need_1	Agile	8	10	12	6	36
Need_1	Mixed	2	7	2	3	14
Need_1	Traditional	1	1	1	0	3
Need_2	Agile	7	13	7	9	36
Need_2	Mixed	1	2	7	3	13
Need_2	Traditional	1	0	0	2	3
Need_3	Agile	2	11	13	9	35
Need_3	Mixed	0	3	9	3	15
Need_3	Traditional	0	1	2	0	3
Need_4	Agile	6	6	15	9	36
Need_4	Mixed	3	5	4	2	14
Need_4	Traditional	0	2	0	1	3
Need_5	Agile	4	10	8	14	36
Need_5	Mixed	2	3	7	3	15
Need_5	Traditional	0	0	1	2	3
Need_6	Agile	5	11	9	12	37

Need_6	Mixed	2	2	5	6	15
Need_6	Traditional	0	0	2	1	3
Need_7	Agile	11	11	11	3	36
Need_7	Mixed	3	4	4	3	14
Need_7	Traditional	0	1	1	1	3
Need_8	Agile	3	12	14	6	35
Need_8	Mixed	3	3	6	2	14
Need_8	Traditional	0	1	0	2	3
Need_9	Agile	5	8	12	10	35
Need_9	Mixed	2	3	8	1	14
Need_9	Traditional	1	0	0	2	3
Need_10	Agile	8	8	10	9	35
Need_10	Mixed	3	7	1	4	15
Need_10	Traditional	0	1	2	0	3
Need_11	Agile	11	13	7	6	37
Need_11	Mixed	4	4	4	3	15
Need_11	Traditional	0	1	1	1	3
Need_12	Agile	4	12	11	8	35
Need_12	Mixed	3	6	5	1	15
Need_12	Traditional	0	0	2	1	3
Need_13	Agile	3	13	10	11	37
Need_13	Mixed	2	4	3	6	15
Need_13	Traditional	0	1	1	1	3

E.2.4 Percentage summary table without NA

statement	Paradigm	total_n	Not needed	Nice to have	Should have	Must have
Need_1	Agile	36	22.22%	27.78%	33.33%	16.67%
Need_1	Mixed	14	14.29%	50.00%	14.29%	21.43%
Need_1	Traditional	3	33.33%	33.33%	33.33%	0.00%
Need_2	Agile	36	19.44%	36.11%	19.44%	25.00%
Need_2	Mixed	13	7.69%	15.38%	53.85%	23.08%
Need_2	Traditional	3	33.33%	0.00%	0.00%	66.67%
Need_3	Agile	35	5.71%	31.43%	37.14%	25.71%
Need_3	Mixed	15	0.00%	20.00%	60.00%	20.00%
Need_3	Traditional	3	0.00%	33.33%	66.67%	0.00%
Need_4	Agile	36	16.67%	16.67%	41.67%	25.00%
Need_4	Mixed	14	21.43%	35.71%	28.57%	14.29%
Need_4	Traditional	3	0.00%	66.67%	0.00%	33.33%
Need_5	Agile	36	11.11%	27.78%	22.22%	38.89%
Need_5	Mixed	15	13.33%	20.00%	46.67%	20.00%
Need_5	Traditional	3	0.00%	0.00%	33.33%	66.67%

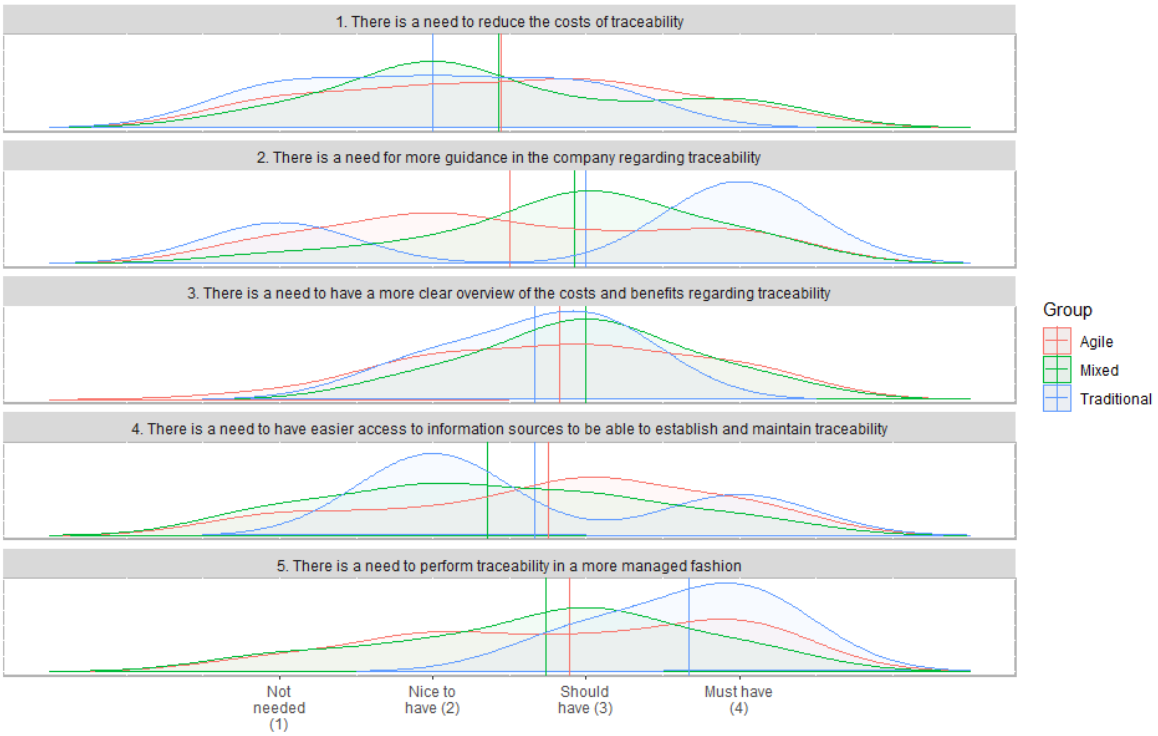
Need_6	Agile	37	13.51%	29.73%	24.32%	32.43%
Need_6	Mixed	15	13.33%	13.33%	33.33%	40.00%
Need_6	Traditional	3	0.00%	0.00%	66.67%	33.33%
Need_7	Agile	36	30.56%	30.56%	30.56%	8.33%
Need_7	Mixed	14	21.43%	28.57%	28.57%	21.43%
Need_7	Traditional	3	0.00%	33.33%	33.33%	33.33%
Need_8	Agile	35	8.57%	34.29%	40.00%	17.14%
Need_8	Mixed	14	21.43%	21.43%	42.86%	14.29%
Need_8	Traditional	3	0.00%	33.33%	0.00%	66.67%
Need_9	Agile	35	14.29%	22.86%	34.29%	28.57%
Need_9	Mixed	14	14.29%	21.43%	57.14%	7.14%
Need_9	Traditional	3	33.33%	0.00%	0.00%	66.67%
Need_10	Agile	35	22.86%	22.86%	28.57%	25.71%
Need_10	Mixed	15	20.00%	46.67%	6.67%	26.67%
Need_10	Traditional	3	0.00%	33.33%	66.67%	0.00%
Need_11	Agile	37	29.73%	35.14%	18.92%	16.22%
Need_11	Mixed	15	26.67%	26.67%	26.67%	20.00%
Need_11	Traditional	3	0.00%	33.33%	33.33%	33.33%
Need_12	Agile	35	11.43%	34.29%	31.43%	22.86%
Need_12	Mixed	15	20.00%	40.00%	33.33%	6.67%
Need_12	Traditional	3	0.00%	0.00%	66.67%	33.33%
Need_13	Agile	37	8.11%	35.14%	27.03%	29.73%
Need_13	Mixed	15	13.33%	26.67%	20.00%	40.00%
Need_13	Traditional	3	0.00%	33.33%	33.33%	33.33%

E.2.5 Mean summary table needs

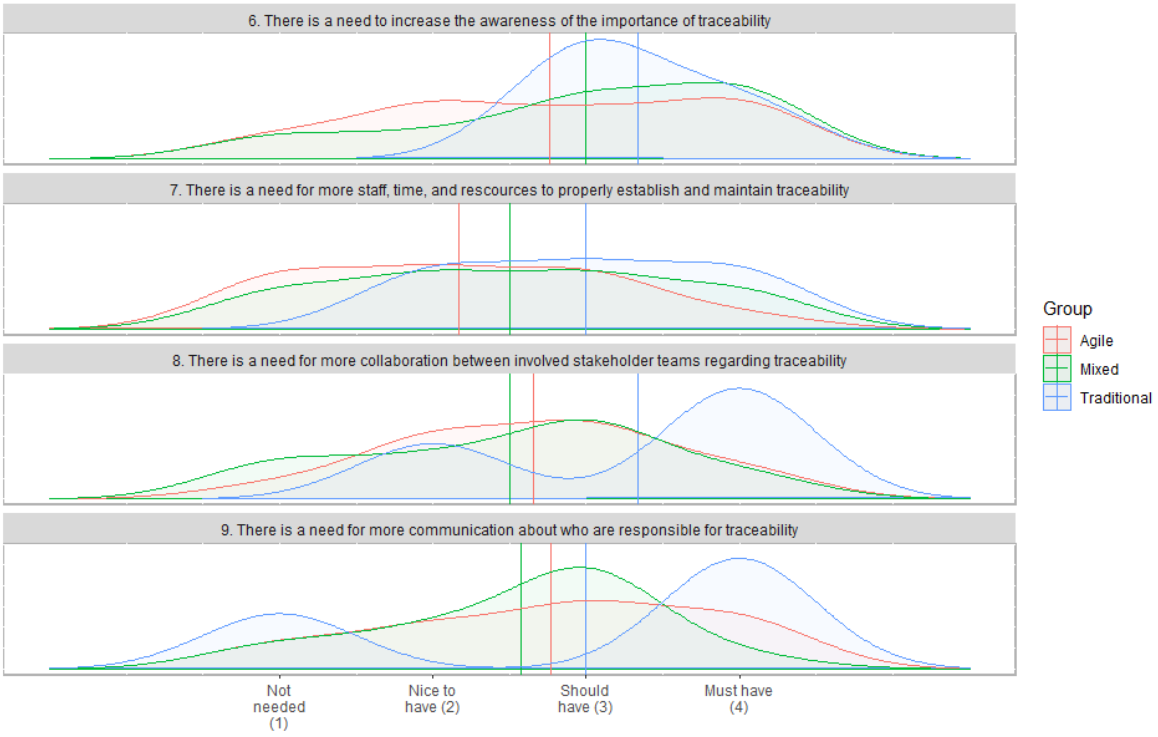
Statement	Agile	Mixed	Traditional	Agile_Mixed	Agile_Traditional	Mixed_Traditional
Need_1	2.44	2.43	2.00	0.02	0.44	0.43
Need_2	2.50	2.92	3.00	-0.42	-0.50	-0.08
Need_3	2.83	3.00	2.67	-0.17	0.16	0.33
Need_4	2.75	2.36	2.67	0.39	0.08	-0.31
Need_5	2.89	2.73	3.67	0.16	-0.78	-0.93
Need_6	2.76	3.00	3.33	-0.24	-0.58	-0.33
Need_7	2.17	2.50	3.00	-0.33	-0.83	-0.50
Need_8	2.66	2.50	3.33	0.16	-0.68	-0.83
Need_9	2.77	2.57	3.00	0.20	-0.23	-0.43
Need_10	2.57	2.40	2.67	0.17	-0.10	-0.27
Need_11	2.22	2.40	3.00	-0.18	-0.78	-0.60
Need_12	2.66	2.27	3.33	0.39	-0.68	-1.07
Need_13	2.78	2.87	3.00	-0.08	-0.22	-0.13

E.2.6 Density graphs

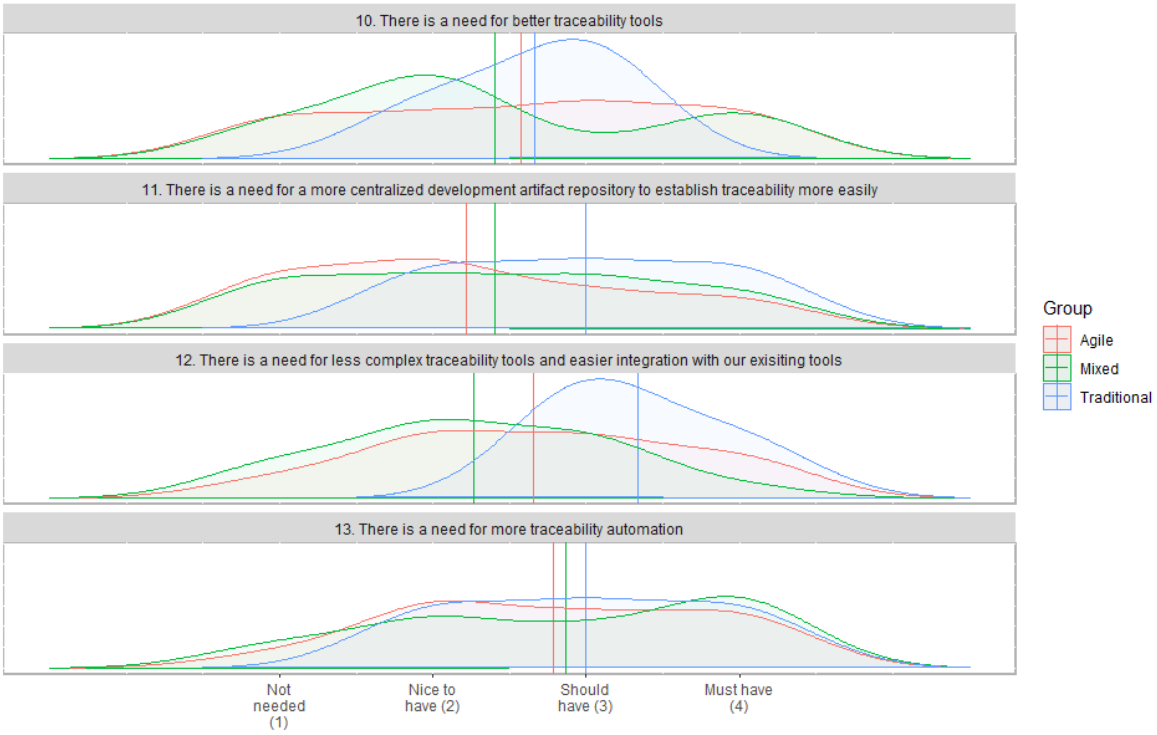
Management need



Social need



Technical need



Appendix F: Safety critical vs non-safety critical descriptive summary tables

F.1 Current situation summary tables

F.1.1 Frequency summary table with NA

statement	Safety_ Critical	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	<NA>	Total
Current_1	No	2	6	13	13	1	35
Current_1	Yes	0	3	10	7	0	20
Current_2	No	1	8	17	4	5	35
Current_2	Yes	2	6	8	3	1	20
Current_3	No	8	15	9	2	1	35
Current_3	Yes	5	5	6	3	1	20
Current_4	No	4	14	12	4	1	35
Current_4	Yes	3	12	3	2	0	20
Current_5	No	4	7	13	8	3	35
Current_5	Yes	8	2	6	3	1	20
Current_6	No	2	4	15	14	0	35
Current_6	Yes	1	1	8	10	0	20
Current_7	No	4	8	12	9	2	35
Current_7	Yes	2	2	9	5	2	20
Current_8	No	4	8	15	5	3	35
Current_8	Yes	2	9	6	3	0	20
Current_9	No	1	10	11	10	3	35

Current_9	Yes	5	4	6	3	2	20
Current_10	No	4	10	10	3	8	35
Current_10	Yes	2	6	9	1	2	20
Current_11	No	5	11	12	6	1	35
Current_11	Yes	3	7	3	4	3	20
Current_12	No	6	11	7	5	6	35
Current_12	Yes	1	9	7	1	2	20
Current_13	No	3	5	11	15	1	35
Current_13	Yes	1	4	8	5	2	20

F.1.2 Percentage summary table with NA

statement	Safety Critical	total_n	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	<NA>
Current_1	No	35	5.71%	17.14%	37.14%	37.14%	2.86%
Current_1	Yes	20	0.00%	15.00%	50.00%	35.00%	0.00%
Current_2	No	35	2.86%	22.86%	48.57%	11.43%	14.29%
Current_2	Yes	20	10.00%	30.00%	40.00%	15.00%	5.00%
Current_3	No	35	22.86%	42.86%	25.71%	5.71%	2.86%
Current_3	Yes	20	25.00%	25.00%	30.00%	15.00%	5.00%
Current_4	No	35	11.43%	40.00%	34.29%	11.43%	2.86%
Current_4	Yes	20	15.00%	60.00%	15.00%	10.00%	0.00%
Current_5	No	35	11.43%	20.00%	37.14%	22.86%	8.57%
Current_5	Yes	20	40.00%	10.00%	30.00%	15.00%	5.00%
Current_6	No	35	5.71%	11.43%	42.86%	40.00%	0.00%
Current_6	Yes	20	5.00%	5.00%	40.00%	50.00%	0.00%
Current_7	No	35	11.43%	22.86%	34.29%	25.71%	5.71%
Current_7	Yes	20	10.00%	10.00%	45.00%	25.00%	10.00%
Current_8	No	35	11.43%	22.86%	42.86%	14.29%	8.57%
Current_8	Yes	20	10.00%	45.00%	30.00%	15.00%	0.00%
Current_9	No	35	2.86%	28.57%	31.43%	28.57%	8.57%
Current_9	Yes	20	25.00%	20.00%	30.00%	15.00%	10.00%
Current_10	No	35	11.43%	28.57%	28.57%	8.57%	22.86%
Current_10	Yes	20	10.00%	30.00%	45.00%	5.00%	10.00%
Current_11	No	35	14.29%	31.43%	34.29%	17.14%	2.86%
Current_11	Yes	20	15.00%	35.00%	15.00%	20.00%	15.00%
Current_12	No	35	17.14%	31.43%	20.00%	14.29%	17.14%
Current_12	Yes	20	5.00%	45.00%	35.00%	5.00%	10.00%
Current_13	No	35	8.57%	14.29%	31.43%	42.86%	2.86%
Current_13	Yes	20	5.00%	20.00%	40.00%	25.00%	10.00%

F.1.3 Frequency summary table without NA

statement	Safety Critical	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree	Total
Current_1	No	2	6	13	13	34
Current_1	Yes	0	3	10	7	20
Current_2	No	1	8	17	4	30
Current_2	Yes	2	6	8	3	19
Current_3	No	8	15	9	2	34
Current_3	Yes	5	5	6	3	19
Current_4	No	4	14	12	4	34
Current_4	Yes	3	12	3	2	20
Current_5	No	4	7	13	8	32
Current_5	Yes	8	2	6	3	19
Current_6	No	2	4	15	14	35
Current_6	Yes	1	1	8	10	20
Current_7	No	4	8	12	9	33
Current_7	Yes	2	2	9	5	18
Current_8	No	4	8	15	5	32
Current_8	Yes	2	9	6	3	20
Current_9	No	1	10	11	10	32
Current_9	Yes	5	4	6	3	18
Current_10	No	4	10	10	3	27
Current_10	Yes	2	6	9	1	18
Current_11	No	5	11	12	6	34
Current_11	Yes	3	7	3	4	17
Current_12	No	6	11	7	5	29
Current_12	Yes	1	9	7	1	18
Current_13	No	3	5	11	15	34
Current_13	Yes	1	4	8	5	18

F.1.4 Percentage summary table without NA

statement	Safety Critical	total_n	Strongly disagree	Somewhat disagree	Somewhat agree	Strongly agree
Current_1	No	34	5.88%	17.65%	38.24%	38.24%
Current_1	Yes	20	0.00%	15.00%	50.00%	35.00%
Current_2	No	30	3.33%	26.67%	56.67%	13.33%
Current_2	Yes	19	10.53%	31.58%	42.11%	15.79%
Current_3	No	34	23.53%	44.12%	26.47%	5.88%
Current_3	Yes	19	26.32%	26.32%	31.58%	15.79%
Current_4	No	34	11.76%	41.18%	35.29%	11.76%
Current_4	Yes	20	15.00%	60.00%	15.00%	10.00%
Current_5	No	32	12.50%	21.88%	40.63%	25.00%
Current_5	Yes	19	42.11%	10.53%	31.58%	15.79%
Current_6	No	35	5.71%	11.43%	42.86%	40.00%
Current_6	Yes	20	5.00%	5.00%	40.00%	50.00%

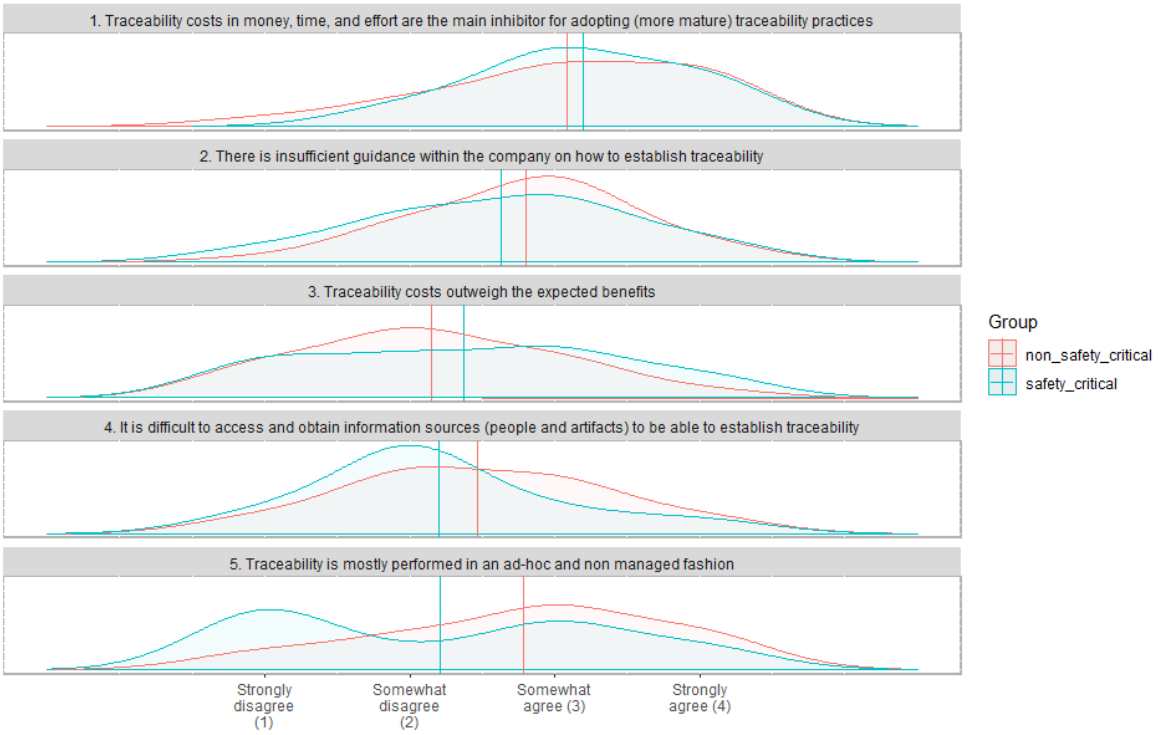
Current_7	No	33	12.12%	24.24%	36.36%	27.27%
Current_7	Yes	18	11.11%	11.11%	50.00%	27.78%
Current_8	No	32	12.50%	25.00%	46.88%	15.63%
Current_8	Yes	20	10.00%	45.00%	30.00%	15.00%
Current_9	No	32	3.13%	31.25%	34.38%	31.25%
Current_9	Yes	18	27.78%	22.22%	33.33%	16.67%
Current_10	No	27	14.81%	37.04%	37.04%	11.11%
Current_10	Yes	18	11.11%	33.33%	50.00%	5.56%
Current_11	No	34	14.71%	32.35%	35.29%	17.65%
Current_11	Yes	17	17.65%	41.18%	17.65%	23.53%
Current_12	No	29	20.69%	37.93%	24.14%	17.24%
Current_12	Yes	18	5.56%	50.00%	38.89%	5.56%
Current_13	No	34	8.82%	14.71%	32.35%	44.12%
Current_13	Yes	18	5.56%	22.22%	44.44%	27.78%

F.1.5 Mean summary table current

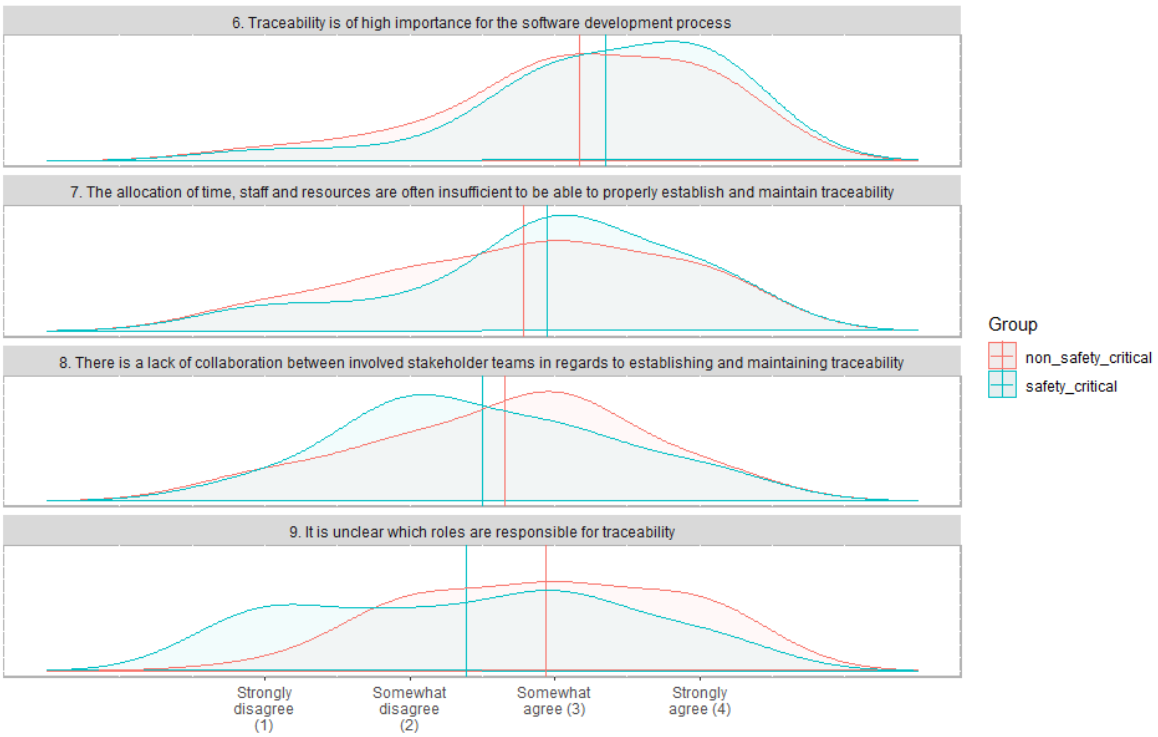
Statement	Non safety	Safety	safety - nsafety
Current_1	3.09	3.20	0.11
Current_2	2.80	2.63	-0.17
Current_3	2.15	2.37	0.22
Current_4	2.47	2.20	-0.27
Current_5	2.78	2.21	-0.57
Current_6	3.17	3.35	0.18
Current_7	2.79	2.94	0.16
Current_8	2.66	2.50	-0.16
Current_9	2.94	2.39	-0.55
Current_10	2.44	2.50	0.06
Current_11	2.56	2.47	-0.09
Current_12	2.38	2.44	0.07
Current_13	3.12	2.94	-0.17

F.1.6 Density graphs

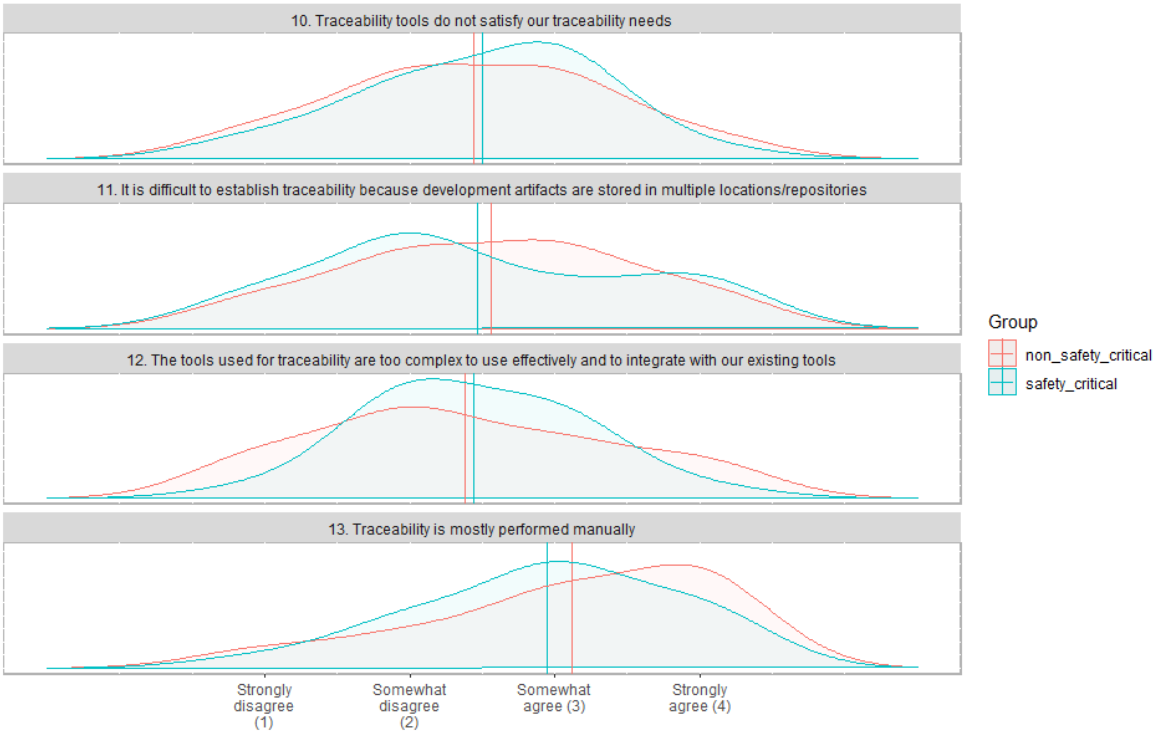
Management current



Social current



Technical current



F.2 Need summary tables

F.2.1 Frequency summary table with NA

statement	Safety Critical	Not needed	Nice to have	Should have	Must have	<NA>	Total
Need_1	No	9	10	11	5	0	35
Need_1	Yes	2	8	4	4	2	20
Need_2	No	6	10	9	9	1	35
Need_2	Yes	3	5	5	5	2	20
Need_3	No	1	11	15	8	0	35
Need_3	Yes	1	4	9	4	2	20
Need_4	No	4	10	15	5	1	35
Need_4	Yes	5	3	4	7	1	20
Need_5	No	5	9	9	12	0	35
Need_5	Yes	1	4	7	7	1	20
Need_6	No	3	9	13	10	0	35
Need_6	Yes	4	4	3	9	0	20
Need_7	No	10	10	12	1	2	35
Need_7	Yes	4	6	4	6	0	20
Need_8	No	2	11	14	5	3	35
Need_8	Yes	4	5	6	5	0	20
Need_9	No	4	8	13	7	3	35
Need_9	Yes	4	3	7	6	0	20
Need_10	No	6	11	7	9	2	35

Need_10	Yes	5	5	6	4	0	20
Need_11	No	9	11	7	8	0	35
Need_11	Yes	6	7	5	2	0	20
Need_12	No	3	12	11	7	2	35
Need_12	Yes	4	6	7	3	0	20
Need_13	No	1	14	10	10	0	35
Need_13	Yes	4	4	4	8	0	20

F.2.2 Percentage summary table with NA

statement	Safety Critical	total_n	Not needed	Nice to have	Should have	Must have	<NA>
Need_1	No	35	25.71%	28.57%	31.43%	14.29%	0.00%
Need_1	Yes	20	10.00%	40.00%	20.00%	20.00%	10.00%
Need_2	No	35	17.14%	28.57%	25.71%	25.71%	2.86%
Need_2	Yes	20	15.00%	25.00%	25.00%	25.00%	10.00%
Need_3	No	35	2.86%	31.43%	42.86%	22.86%	0.00%
Need_3	Yes	20	5.00%	20.00%	45.00%	20.00%	10.00%
Need_4	No	35	11.43%	28.57%	42.86%	14.29%	2.86%
Need_4	Yes	20	25.00%	15.00%	20.00%	35.00%	5.00%
Need_5	No	35	14.29%	25.71%	25.71%	34.29%	0.00%
Need_5	Yes	20	5.00%	20.00%	35.00%	35.00%	5.00%
Need_6	No	35	8.57%	25.71%	37.14%	28.57%	0.00%
Need_6	Yes	20	20.00%	20.00%	15.00%	45.00%	0.00%
Need_7	No	35	28.57%	28.57%	34.29%	2.86%	5.71%
Need_7	Yes	20	20.00%	30.00%	20.00%	30.00%	0.00%
Need_8	No	35	5.71%	31.43%	40.00%	14.29%	8.57%
Need_8	Yes	20	20.00%	25.00%	30.00%	25.00%	0.00%
Need_9	No	35	11.43%	22.86%	37.14%	20.00%	8.57%
Need_9	Yes	20	20.00%	15.00%	35.00%	30.00%	0.00%
Need_10	No	35	17.14%	31.43%	20.00%	25.71%	5.71%
Need_10	Yes	20	25.00%	25.00%	30.00%	20.00%	0.00%
Need_11	No	35	25.71%	31.43%	20.00%	22.86%	0.00%
Need_11	Yes	20	30.00%	35.00%	25.00%	10.00%	0.00%
Need_12	No	35	8.57%	34.29%	31.43%	20.00%	5.71%
Need_12	Yes	20	20.00%	30.00%	35.00%	15.00%	0.00%
Need_13	No	35	2.86%	40.00%	28.57%	28.57%	0.00%
Need_13	Yes	20	20.00%	20.00%	20.00%	40.00%	0.00%

F.2.3 Frequency summary table without NA

statement	Safety Critical	Not needed	Nice to have	Should have	Must have	Total

Need_1	No	9	10	11	5	35
Need_1	Yes	2	8	4	4	18
Need_2	No	6	10	9	9	34
Need_2	Yes	3	5	5	5	18
Need_3	No	1	11	15	8	35
Need_3	Yes	1	4	9	4	18
Need_4	No	4	10	15	5	34
Need_4	Yes	5	3	4	7	19
Need_5	No	5	9	9	12	35
Need_5	Yes	1	4	7	7	19
Need_6	No	3	9	13	10	35
Need_6	Yes	4	4	3	9	20
Need_7	No	10	10	12	1	33
Need_7	Yes	4	6	4	6	20
Need_8	No	2	11	14	5	32
Need_8	Yes	4	5	6	5	20
Need_9	No	4	8	13	7	32
Need_9	Yes	4	3	7	6	20
Need_10	No	6	11	7	9	33
Need_10	Yes	5	5	6	4	20
Need_11	No	9	11	7	8	35
Need_11	Yes	6	7	5	2	20
Need_12	No	3	12	11	7	33
Need_12	Yes	4	6	7	3	20
Need_13	No	1	14	10	10	35
Need_13	Yes	4	4	4	8	20

F.2.4 Percentage summary table without NA

statement	Safety Critical	total_n	Not needed	Nice to have	Should have	Must have
Need_1	No	35	25.71%	28.57%	31.43%	14.29%
Need_1	Yes	18	11.11%	44.44%	22.22%	22.22%
Need_2	No	34	17.65%	29.41%	26.47%	26.47%
Need_2	Yes	18	16.67%	27.78%	27.78%	27.78%
Need_3	No	35	2.86%	31.43%	42.86%	22.86%
Need_3	Yes	18	5.56%	22.22%	50.00%	22.22%
Need_4	No	34	11.76%	29.41%	44.12%	14.71%
Need_4	Yes	19	26.32%	15.79%	21.05%	36.84%
Need_5	No	35	14.29%	25.71%	25.71%	34.29%
Need_5	Yes	19	5.26%	21.05%	36.84%	36.84%
Need_6	No	35	8.57%	25.71%	37.14%	28.57%
Need_6	Yes	20	20.00%	20.00%	15.00%	45.00%
Need_7	No	33	30.30%	30.30%	36.36%	3.03%

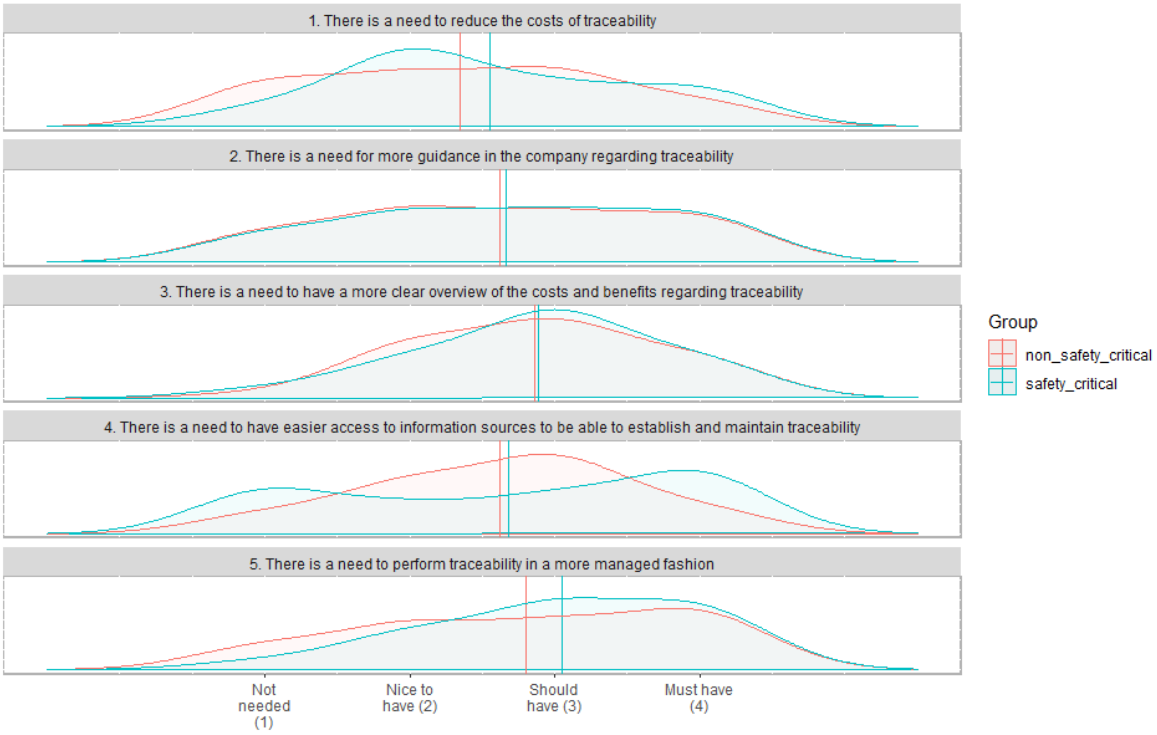
Need_7	Yes	20	20.00%	30.00%	20.00%	30.00%
Need_8	No	32	6.25%	34.38%	43.75%	15.63%
Need_8	Yes	20	20.00%	25.00%	30.00%	25.00%
Need_9	No	32	12.50%	25.00%	40.63%	21.88%
Need_9	Yes	20	20.00%	15.00%	35.00%	30.00%
Need_10	No	33	18.18%	33.33%	21.21%	27.27%
Need_10	Yes	20	25.00%	25.00%	30.00%	20.00%
Need_11	No	35	25.71%	31.43%	20.00%	22.86%
Need_11	Yes	20	30.00%	35.00%	25.00%	10.00%
Need_12	No	33	9.09%	36.36%	33.33%	21.21%
Need_12	Yes	20	20.00%	30.00%	35.00%	15.00%
Need_13	No	35	2.86%	40.00%	28.57%	28.57%
Need_13	Yes	20	20.00%	20.00%	20.00%	40.00%

F.2.5 Mean summary table needs

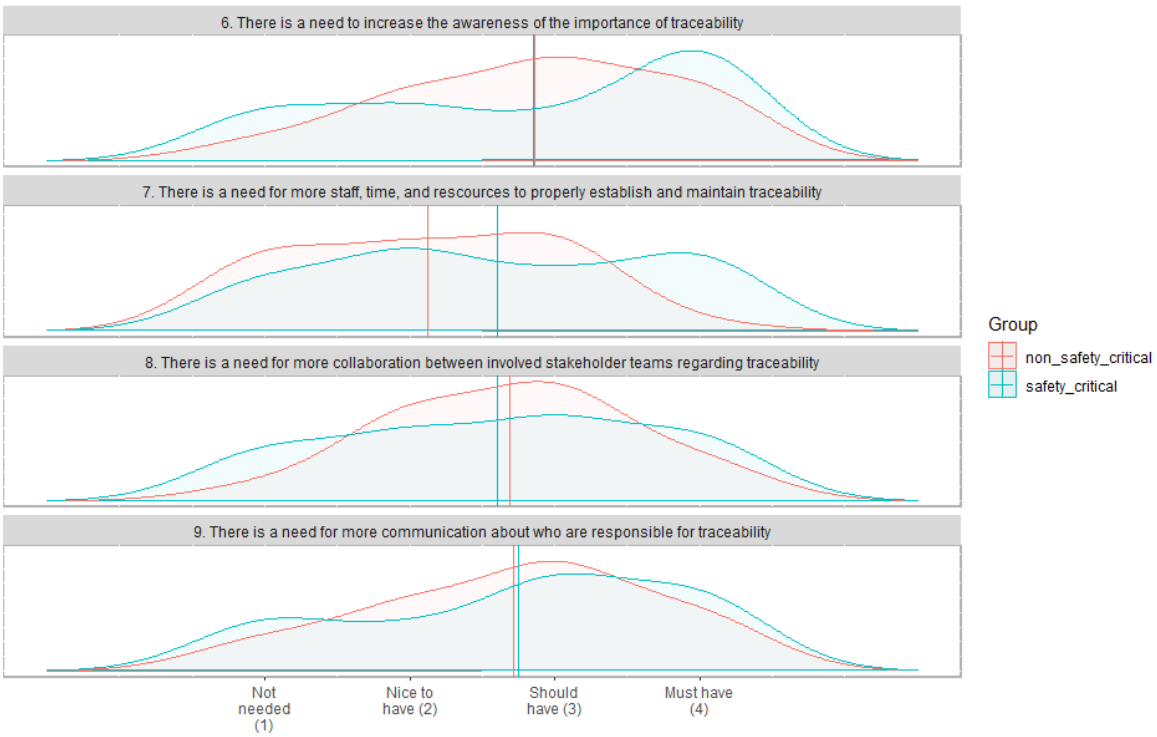
Statement	Non safety	Safety	safety - nsafety
Need_1	2.34	2.56	0.21
Need_2	2.62	2.67	0.05
Need_3	2.86	2.89	0.03
Need_4	2.62	2.68	0.07
Need_5	2.80	3.05	0.25
Need_6	2.86	2.85	-0.01
Need_7	2.12	2.60	0.48
Need_8	2.69	2.60	-0.09
Need_9	2.72	2.75	0.03
Need_10	2.58	2.45	-0.13
Need_11	2.40	2.15	-0.25
Need_12	2.67	2.45	-0.22
Need_13	2.83	2.80	-0.03

F2.6 density graphs

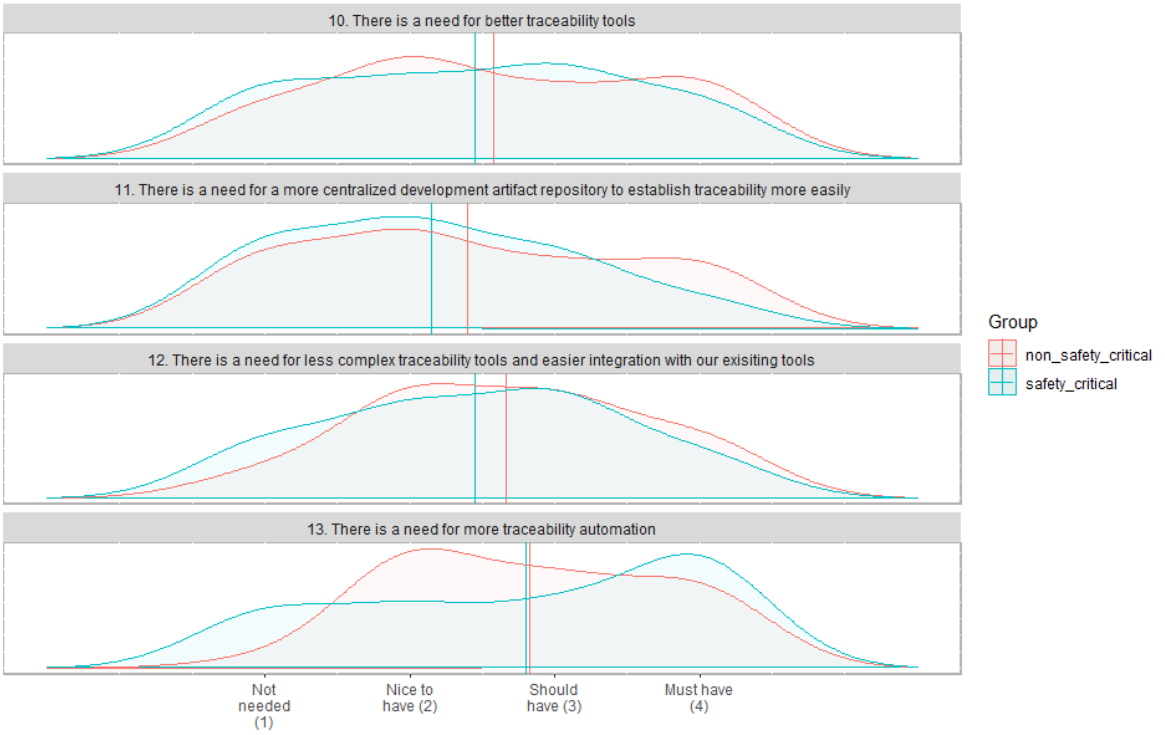
Management need



Social need



Technical need


10. There is a need for better traceability tools
11. There is a need for a more centralized development artifact repository to establish traceability more easily
12. There is a need for less complex traceability tools and easier integration with our exisiting tools
13. There is a need for more traceability automation
Not needed (1)
Nice to have (2)
Should have (3)
Must have (4)
Group
non_safety_critical
safety_critical

Appendix G: Statement literature mapping

Literature / statements	1.1	1.2	2.1	2.2	3.1	3.2	4.1	4.2	5.1	5.2	6.1	6.2	7.1	7.2	8.1	8.2	9.1	9.2	10.1	10.2	11.1	11.2	12.1	12.2	13.1	13.2
Regan et al., (2012)	x	x	x	x	x	x	x	x			x	x					x	x			x	x	x	x		
(Ramesh, 1998)			x	x							x	x	x	x												
(Arkley & Riddle, 2005)					x	x																				
(Cleland-Huang, 2006, 2012)			x	x							x	x													x	x
(Mäder et al., 2009)			x	x															x	x						
(Blaauboer et al., 2007)											x	x														
(Bouillon et al., 2013)					x	x																				
(Wohlrab et al., 2018)						x									x	x										
(O. C. Z. Gotel & Finkelstein, 1994)							x	x													x	x				
(Cleland-Huang et al., 2014)									x	x																
(Orlena Gotel et al., 2012)	x	x									x	x							x	x					x	x
(Egyed, Grunbacher, Heindl, & Biffl, 2007)	x	x																								
(Antoniol et al., 2017)							x	x													x	x				

Appendix H: Validation feedback form req lab, MBI students, and native English speaker

Validation form: Software traceability

Thank you for participating in the validation of our questionnaire regarding software traceability.

Name:

Comments on design/layout

Comments on content/grammar

Any tips or other feedback to improve the questionnaire

Appendix I: Correlation analysis

I.1 Correlation analysis: current situation.

Current statements correlation

Statement	1	2	3	4	5	6	7	8	9	10	11	12	13
1	1.000	0.180	0.157	0.224	0.267	0.128	0.362	0.049	0.206	0.290	0.285	0.260	0.112
2	0.180	1.000	0.164	0.379	0.433	0.220	0.355	0.340	0.348	0.469	0.398	0.486	0.278
3	0.157	0.164	1.000	0.094	0.240	0.390	0.214	0.037	0.159	0.165	0.059	0.135	0.066
4	0.224	0.379	0.094	1.000	0.214	0.321	0.332	0.214	0.229	0.300	0.208	0.394	0.421
5	0.267	0.433	0.240	0.214	1.000	0.396	0.418	0.266	0.366	0.194	0.296	0.201	0.415
6	0.128	0.220	0.390	0.321	0.396	1.000	0.265	0.118	0.114	0.007	0.135	0.041	0.090
7	0.362	0.355	0.214	0.332	0.418	0.265	1.000	0.431	0.288	0.461	0.083	0.229	0.558
8	0.049	0.340	0.037	0.214	0.266	0.118	0.431	1.000	0.384	0.321	0.185	0.256	0.424
9	0.206	0.348	0.159	0.229	0.366	0.114	0.288	0.384	1.000	0.220	0.063	0.288	0.225
10	0.290	0.469	0.165	0.300	0.194	0.007	0.461	0.321	0.220	1.000	0.539	0.613	0.456
11	0.285	0.398	0.059	0.208	0.296	0.135	0.083	0.185	0.063	0.539	1.000	0.559	0.240
12	0.260	0.486	0.135	0.394	0.201	0.041	0.229	0.256	0.288	0.613	0.559	1.000	0.304
13	0.112	0.278	0.066	0.421	0.415	0.090	0.558	0.424	0.225	0.456	0.240	0.304	1.000

Table: Correlation of current statements. Blue denote positive correlations, in which the palette goes from light blue to dark blue – low correlation to high correlation. Red denote negative correlations, in which the palette goes from light red to dark red – low correlation to high correlation.

I.2 Correlation analysis: needs

Need statements correlations

Statement	1	2	3	4	5	6	7	8	9	10	11	12	13
1	1.000	0.196	0.124	0.216	0.284	0.011	0.153	0.065	0.096	0.429	0.450	0.209	0.281
2	0.196	1.000	0.409	0.231	0.487	0.639	0.600	0.337	0.576	0.437	0.436	0.302	0.375
3	0.124	0.409	1.000	0.360	0.425	0.600	0.445	0.312	0.483	0.351	0.340	0.181	0.293
4	0.216	0.231	0.360	1.000	0.279	0.324	0.254	0.369	0.450	0.423	0.532	0.329	0.484
5	0.284	0.487	0.425	0.279	1.000	0.701	0.441	0.569	0.369	0.369	0.397	0.165	0.452
6	0.011	0.639	0.600	0.324	0.701	1.000	0.596	0.471	0.550	0.391	0.397	0.189	0.314
7	0.153	0.600	0.445	0.254	0.441	0.596	1.000	0.247	0.419	0.314	0.331	0.247	0.177
8	0.065	0.337	0.312	0.369	0.569	0.471	0.247	1.000	0.473	0.257	0.260	0.208	0.412
9	0.096	0.576	0.483	0.450	0.369	0.550	0.419	0.473	1.000	0.516	0.440	0.407	0.411
10	0.429	0.437	0.351	0.423	0.369	0.391	0.314	0.257	0.516	1.000	0.579	0.695	0.497
11	0.450	0.436	0.340	0.532	0.397	0.397	0.331	0.260	0.440	0.579	1.000	0.457	0.469
12	0.209	0.302	0.181	0.329	0.165	0.189	0.247	0.208	0.407	0.695	0.457	1.000	0.498
13	0.281	0.375	0.293	0.484	0.452	0.314	0.177	0.412	0.411	0.497	0.469	0.498	1.000

Table: Correlation of need statements. Blue denote positive correlations, in which the palette goes from light blue to dark blue – low correlation to high correlation. Red denote negative correlations, in which the palette goes from light red to dark red – low correlation to high correlation.