CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Traceability Link Correctness: Project-Specific or Generic?

Master's thesis in Software Engineering and Management

Sanja Colak & Mayra Nilsson

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2021

Master's thesis 2021

Traceability Link Correctness: Project-Specific or Generic?

Sanja Colak & Mayra Nilsson

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

Traceability Link Correctness: Project-Specific or Generic?
Sanja Colak & Mayra Nilsson

Supervisor: Jan-Philipp Steghöfer, Software Engineering Division
Examiner: Jennifer Horkoff, Software Engineering Division

Master's Thesis 2021
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2021

Sanja Colak & Mayra Nilsson
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

To enable proper maintenance of a software product, it is important to discover new and maintain existing traceability links between artifacts. Correctness of trace links is important as it builds trust in existing trace links and improves software maintenance. Neglecting traceability in a project has negative impact on the software quality and increases project development cost and time. Additionally, properly set trace links in a project provide flexibility within a development team as the knowledge of understanding inter-dependencies in the system does not rely on a domain expert only. The research community opinion on trace link correctness differs, which results in contradictory solutions on how to evaluate trace link correctness. In this paper, we identify notions of trace link correctness applicable on the data model level and examine if the uses of the defined notions are generic or project-specific. Additionally, the paper examines if the evaluation of trace links using the defined notions requires a domain expert or if they can be evaluated by software engineers lacking domain expertise.
The study is conducted in two iterations. The first iteration focuses on identifying notions of trace link correctness and examines if they are generic or project-specific. The second iteration focuses on understanding if the evaluation of trace link correctness using the identified notions requires a domain expert. Five notions of trace link correctness are identified: Versioning, Lifetime/Lifespan, Non-Duplicated Trace Links, Unique Artifact Identification and Mandatory Artifacts & and Mandatory Trace Link. Non-Duplicated Trace Links and Unique Artifact Identification are identified as generic while Versioning, Lifetime/Lifespan and Mandatory Artifacts & and Mandatory Trace Link are identified as project-specific. Furthermore, it was found that not only a domain expert, but also experienced software engineers can evaluate trace link correctness using the identified notions.

Keywords: Traceability, trace links, trace link correctness, Traceability Information Model, Data Model, project-specific.

Acknowledgements

First of all, we would like to thank to our supervisor Jan-Philipp Steghöfer from the Computer Science and Engineering department at the University of Gothenburg, for guiding and navigating us through the study. Without his patience, time involved in reading the paper all over again and providing us the constructive feedback we would not be able to conduct this research.

Secondly, we would also like to thank to our supervisor for providing us contact for the domain experts interviewed in this research. Without domain experts we would not be able to derive the research in the way we did.

Would like to thank to our academic examiner, Jennifer Horkoff, from the Computer Science and Engineering department at the University of Gothenburg, for providing us a constructive feedback during the mid-term evaluation of the thesis work.

Big thank to our opponent for reading our work and providing us with her valuable feedback.

Additionally, we would like to thank to our participants in this study for providing us valuable answers in the interviews and questionnaires which was core of this research.

Finally, we would like to thank to our families for giving us unconditional support and motivation during the entire study period. Without them it would be almost impossible to finish this journey. We are very proud to have them in our lives.

Sanja Colak & Mayra Nilsson, Gothenburg, June 2021

Contents

List of Figures

List of Tables

1
Introduction

Software traceability can be defined in terms of the ability to trace and follow the life of an artifact through its entire life cycle [1]. Software traceability is a key activity in software maintenance as it refers to the process of discovering and maintaining links between software artifacts, for example how a software requirement relates to source code or a test case [2]. As traceability is often created and maintained by humans, there is a risk of making mistakes [1]. This could potentially affect the quality of existing traceability links negatively. Traceability and traceability link "quality" is usually presented in measurable properties: correctness, completeness, accuracy, confidence, etc [3]. This study focused on the property of correctness of trace links. As previous studies have shown, this is an important property of trace link quality [4].

Correctness of trace links is important as it builds trust in existing links and increases the likelihood that trace links will be used and improved over time [3]. It is therefore important that stakeholders understand the current state of correctness of trace links in order to assess whether or not they can trust the existing trace data. Cleland-Huang et al. stated that developing techniques for assessing and communicating the current state of traceability in a project is a means to achieve this understanding and thereby building trust [5]. A high rate of incorrect trace links can also have the adverse effect of leading to project failures, cost and schedule overruns and sub-optimal designs [6]. Previous studies have shown that neglecting traceability and trace link correctness in a project leads to a decrease of the system quality and causes additional revisions, thereby increasing project development time and cost [7]. It also leads to a loss of knowledge if key individuals leave the project, as properly set up trace links support other team members in understanding interdependencies in the system [5, 7].

The research community has different views on trace link correctness. This results in different and sometimes contradictory proposed solutions on how to evaluate trace link correctness. In several papers the definition of correctness is defined by domain experts. Maro et al. state that correctness of traceability links is manually determined by consulting a domain expert [3]. In a similar fashion, De Lucia et al. use a set of trace links provided by domain experts and this set is treated as the absolute truth [1]. Also, many researchers talk about calculating precision and recall to evaluate correctness of candidate trace links, but the determination of the correctness is also based on human judgement [8, 9, 10, 11]. However, none of them mention what notions of correctness are used in order to define what correctness

means, neither what the definition of trace link correctness is.

Trace link correctness can be considered on two levels. These levels are the Information Model Level and the Data Model Level. The traceability life cycle comprises the activities planning and managing a traceability strategy, creating and maintaining traceability data and using the traceability data [12]. The traceability planning strategy has a Traceability Information Model (TIM) as an output, which defines types of artifacts to be traced and their relations [13]. The TIM is project-specific and represents the project's traceability concerns specifying artifact classes and permissible trace links [13]. This is the upper level of trace data, i.e. the Information Model Level. On the lower level, the Traceability Data Model (TDM) represents all traceability data created and maintained throughout the development life cycle. The TDM consists of all instanced artifacts, trace links and trace paths [12]. This represents the Data Model Level.

This research defines the notions of correctness on two different levels:
- Notions of correctness at the Information Model Level, which deals with the definition of the trace link types defined in the project TIM.
- Notions of correctness on the Data Model Level, which deals with the definition of the instances of permissible trace links from the TIM in the project.

This paper focuses on investigating and understanding correctness of trace links on the Data Model Level. In the initial study phase, notions identified as belonging to both the Information Model Level and Data Model Level were included. This in order to ensure that even if a notion was found to be applicable on the Information Model Level by literature, the study would confirm if this was the case in practice or if it could potentially be used on the Data Model Level as well, thereby extending the knowledge in the traceability field. Furthermore, the study evaluates if domain expertise is required to evaluate trace link correctness based on the identified notions of trace link correctness. The knowledge gained will support practitioners to select what notions to use in a specific project depending on the availability of domain experts.

The research questions we aim to answer in this research are:
RQ1: What notions define trace link correctness?
RQ2: Are the defined notions of trace links correctness generic or project-specific?
RQ3: Does evaluation of trace link correctness based on the identified notions require a domain expert?

RQ1 focuses on discovering notions of trace link correctness. Various scientific articles and books provided data on the notions of trace link correctness currently identified in the field of traceability. This resulted in a list of notions of trace link correctness where each notion was defined based on their usage in the literature.

The focus of RQ2 is to understand whether or not the identified notions of trace

link correctness can be used generically for any project, or if their use must be customized according to project needs making them project-specific.

RQ3 focuses on investigating if the evaluation of trace link correctness based on the identified notions requires a domain expert, if trace link correctness can be evaluated by a software engineer without having deeper knowledge about the domain. This was investigated to understand if the common practice used in literature, i.e., evaluation of trace link correctness by domain experts, is a requirement for the identified notions.

This study aims to bring new knowledge into the research community on what trace link correctness is and how it should be treated. Currently, studies either use the assistance of domain experts or fall back on assumptions fitting the study. This results in all studies being of limited usefulness outside the study parameters. The study gathers various notions of correctness and tests if they are project-specific. This aids practitioners in the field and future research work by providing additional knowledge on how to tackle the notion of correctness when developing new systems or performing studies in the traceability field.

The paper is organised as follows: Chapter 2 details the background and review literature researching the same area, Chapter 3 presents the way in which the study was conducted, Chapter 4 outlines the results of the data analysis, results which are then discussed in Chapter 5, and finally, Chapter 6 summarises the content of the paper.

2
Background

This chapter presents the background information and existing literature in the field of software traceability, traceability links and existing notions of trace link correctness.

2.1 Traceability and its importance

Traceability between various artifacts is an essential element of the software development process as it is deemed to assist practitioners in comprehension, efficient development as well as in the effective management of software systems [14]. According to Lago et al. traceability is the ability to describe and follow the life of software artifacts which is represented by the links that connect the related artifacts [15]. The Center of Excellence for Software Traceability (CoEST) defines a trace link as the "specified association/relation between a pair of artifacts, one comprising the source artifact (e.g., tests) and one comprising the target artifact (e.g., code)"[1]. Furthermore, artifact relationships are represented as trace links while link endpoints represent different views of artifacts [16].

Despite the importance of traceability, the task of creating and maintaining correct and accurate trace links can become challenging as the number of links grows at the same pace that the system evolves. Engineers are often forced to manually find and review regulatory documents to find required trace links in a project. Different research studies have shown that badly managed trace links can potentially lead to project failures and budget overruns [17]. Moreover, inadequate traceability contributes to less maintainable software and more defects due to inconsistencies [17]. Well maintained trace links increase the reliability and maintainability of software systems. Therefore traceability is not only important for software artifacts but also has been adopted as a major component of many standards for software development such as ISO 9001:200, IEEE Standard 92, CMMI, etc [18].

Traceability links have intrinsic components such as source and targets. Their elements indicate various software artifacts and have several properties such as artifact name, artifact type (e.g requirement, design diagram, test case, etc), location, version, etc. In addition to the properties of the source and the target, a link can also have several general properties such as version, rationale description and status. Some of these properties are independent while others depend on specific modifica-

[1]http://sarec.nd.edu/coest

tions [15]. To better understand traceability, researchers have investigated the use of automated techniques to identify crucial links. However establishing, validating and maintaining trace links is often expensive and time consuming [18]. To address these problems, previous research focused on tracing documentation to code [19] or design specification to code, paying limited attention to tracing between artifacts [20].

Traceability can be setup as a project-specific Traceability Information Model (TIM) to be used within a project. This model defines what artifacts can be created within a project and also how they shall relate to each other by defining permissible trace links and trace paths. However, it has been found that practitioners rarely see the benefit of creating such information models and therefore they are rarely defined and used [13]. Traceability information models are not a new concept [13]. The Traceability Data Model (TDM) is the level this study focuses on as some scoping was required due to time constraints of the study. It was judged as not feasible to consider also the Traceability Information Model (TIM) level within the given timeframe. The TDM level represents traceability data that is created, maintained and used throughout a development life cycle [12]. The relation between the two model levels can be explained as the information model defining what elements can exist in trace data of a project while the data model contains the instances of these elements [12].

2.2 Related work

In terms of cost and time, trace links are expensive to create but more challenging is to accurately maintain the links as the system evolves [**?**]. Previous studies shown that traceability data often suffers from non-completed information such as missing artifacts, missing trace paths, redundant trace paths and missing trace links [21, 14]. Trace links can be created and maintained manually, semi-automaticlly and fully-automaticlly [4].

Maro et al. define a consistency function which evaluates validity, completeness and correctness of the TDM [3]. Validity is defined as the requirement that all trace links in the TDM must conform to their corresponding permissible links in the TIM. Completeness focuses on coverage of requirements, i.e. how many requirements have a trace link connected. Correctness is evaluated manually by a domain expert. In the study, a Traceability Maintainer is also defined. The Traceability Maintainer is proposed to automatically delete broken trace links, i.e. delete trace links leading to deleted artifacts. It is also proposed to identify trace links connected to old versions of modified artifacts. This is detected by using and automatic Version Control System (VCS) which allows the Traceability Maintainer to detect when a new version of an artifact has been released. These links are proposed to be manually reviewed by a domain expert.

Cleland-Huang et al. discuss evaluation of trace link correctness with regards to trace maintenance and trace integrity [5]. Trace maintenance is identified as essen-

tial as trace links become stale if they are not evolved when connected artifacts are modified. Automatic evolution of trace links is considered as a difficult area to automate. Trace integrity is concerned with validation of trace link correctness, which often involves a human feedback and analysis [5]. It is noted that humans have different strategies when examining trace links, such as only reviewing a few random links and accepting all if the random few were good, which results in approximately 25% incorrect feedback. Furthermore, there are several automated traceability tools such as ADAMS [22], POIROT [23] and RETRO [24], which integrate human feedback in order to determine trace link correctness. Three main areas of work regarding automatic trace link evaluation were identified in the study. These were automatic evaluation based on semantics of a trace, using trace patterns between artifacts to identify missing links and validate consistency of existing links and the third area being using metrics such as coverage and completeness to evaluate quality of trace link data.

Rahimi et al. explore a problem of trace link evolution across multiple software versions of safety-critical systems [25]. The focus of the work is on evolving trace links between multiple artifacts, such as source code, requirements and safety-critical artifacts. In the research, algorithms that support trace link evolution across a diverse set of artifacts to achieve lightweight maintenance of trace links are proposed. The approach used identifies high-level change scenarios and specifies them in terms of low-level classes of change. The research specifies properties that hold before and after each change in the artifact, devise ways to measure properties in order to detect a change event and define trace link evolution heuristic for each change scenarios in order to specify links that need to be updated or changed. The algorithm created is implemented in a tool referred to as the Trace Link Evolver (TLE). The experiment to support the research showed that the tool was able to evolve trace links across 27 versions of the system used for this purpose. Also, results showed very high accuracy in evolution of trace links. The accuracy of the tool was measured by analysing the results of the tool manually by three members of the research group and each incorrect evolution was further analysed to understand why the tool had evolved a trace link which should have remained unchanged.

Rempel et al. clearly distinguished different phases in the traceability process in their research [12]. These phases are divided as follows: planning and managing a traceability strategy, creating and maintaining traceability data and using traceability data. According to the research, the planning phase refers to the specification of the traceable artifact classes and permissible trace link classes. The output of this phase is the Traceability Information Model (TIM). The main elements of the TIM are artifact classes which specify artifacts that traceability is required for, permissible trace link classes which represents a relationship between these artifacts and trace path classes which are a representation of a sequence of trace links between artifacts. The Traceability Data Model (TDM) refers to all traceability data created, maintained and used during the development life cycle. In a TDM, an artifact is defined as an output that is produced and maintained throughout the software life cycle.

Mäder et al. identified common traceability problems based on evaluation of traceability information for ten submissions prepared by manufactures for review at the US FDA [14]. Various remedies were proposed to the identified issues [14]. Six recommended practices for strategic traceability are defined, and two of them directly relate to defining trace link correctness. It is suggested that each project should have a unique TIM to be used when creating Requirement Trace Matrices (RTM). The TIM ensures that developers follow the project trace strategy and reduces the risk of missing important traces [14]. Furthermore, each artifact in the system is suggested to have a unique ID which enables automatic detection of redundant trace links and avoids misunderstanding during communication between development engineers. The research also suggests that trace link granularity must be clearly defined in the TIM and RTMs and must be periodically evaluated in order to ensure that trace links are created at the correct granularity [14]. Paech et al. describe granularity as "different levels of traceability" and they argue that the correct granularity level depends on the specific purpose that should be supported [26]. The level of granularity is usually user defined [14, 17, 27], Ghabi et al. found that not defining the level of granularity for trace links to be generated negatively affects quality of generated trace links [17]. The research showed a 16% increase of incorrect trace links generated when not defining granularity. They concluded that the granularity level to choose depends on the need of the stakeholders [17]. Mäder et al. also identified a problem of duplicated trace information. This occurs in two forms: when identical trace links are included multiple times in the trace matrix and the second form happens when similar traces are established at different levels of granularity [14]. To prevent the first form from occurring, Mäder et al. suggest storing trace links in a database-like repository [14]. In this case, executing trace queries would find duplicates enabling removal of them. To support detection of duplicate, the importance of having unique identification of artifacts is highlighted [14]. It is stated that as a fundamental principle of traceability, each artifact must have a unique identifier. Mäder et al. suggests using prefixes to distinguish unique artifact types in the TIM and these prefixes should be intuitive to stakeholders [14]. The prefix is then to be used as part of the unique identifier of the instanced artifact in the TDM.

Rempel et al. developed an approach which parses guidelines for safety-critical systems to extract a Traceability Information Model depicting software artifact types and their prescribed traces [21]. The approach then analyzes the traceability data within a project finding missing traceability paths, redundant or inconsistent data, incorrect artifact links and other issues found are highlighted. If guidelines or regulations which require specific trace paths between artifacts are used in a project, then the project traceability data must contain trace links covering those paths. This includes the artifacts and involved trace links building up the trace path. Seven projects were analyzed using the approach and none were found to fully conform to the relevant guidelines.

The research community is tackling many different issues on traceability and no coherent view and approach exists on what makes a trace link correct. Therefore

this study aims to expand the knowledge base by providing a set of notions that can be used for defining trace link correctness for future studies.

3
Research Methodology

3.1 Design methods and procedures

This research was organized as Design Science Research. As the design science approach is oriented towards creating a successful artifact intended to solve identified problems [28, 29], we found it suitable for this research. As the Design Science approach supports qualitative and quantitative evaluations of the developed artifact, it allows the opportunity to apply both empirical and qualitative methods [29].

Peffers et al. propose the following structure of the DSR: (1) Problem identification and motivation, (2) Objectives of a solution, (3) Design and development, (4) Demonstration, (5) Evaluation and (6) Communication [28] as shown in Figure 3.1 below.

The study was divided in two iterations, each following the steps outlined above. In this research, the first, second and third steps are merged into a Preparation phase, while Demonstration, Evaluation and Communication phases remain unchanged, as shown in Figure 3.2 below.

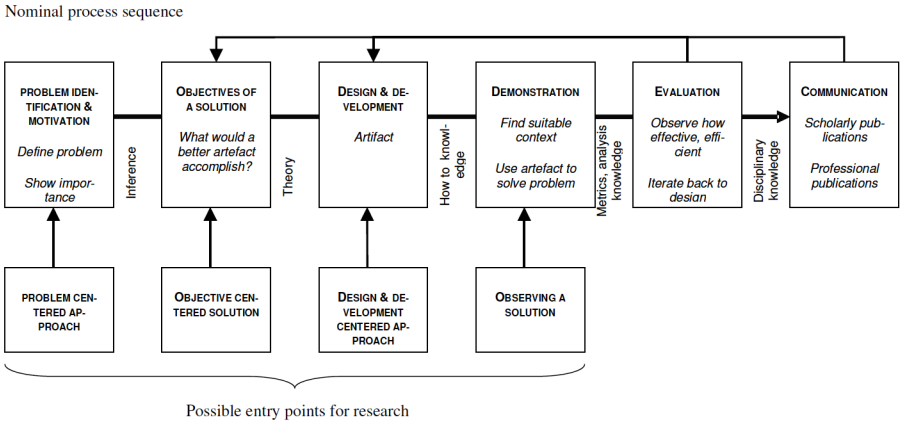

Figure 3.1: Design Science Research process (DSRP) model

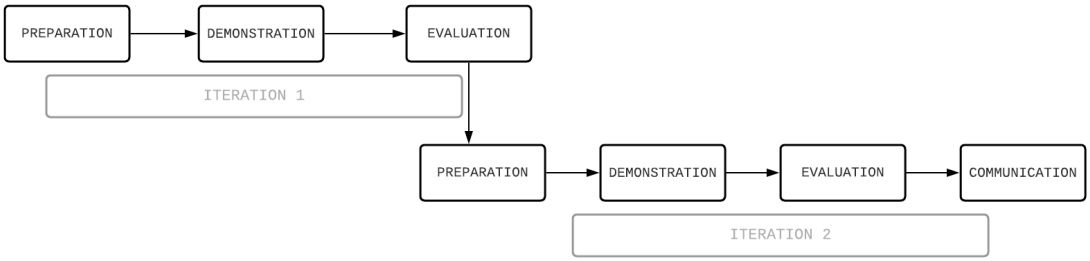

Figure 3.2: Design Science Research process used in this research

The preparation phase for each iteration comprised artifact creation and artifact update and preparation of demonstration phase. Demonstration phase covered data collection process, while Evaluation phase focused on analysis of the collected data. The last objective of the research was to communicate the problem and why it was important to bring it up to society by answering the research questions stated in Section 1. Furthermore, the conclusion elaborates on how the proposed solution of this study contributes to solving the identified problem and identifies future work needed to further refine the knowledge gained. We communicated the results of the research as the final step after concluding the work. The outcome of the research was communicated in the form of a written report and a presentation.

The study took a qualitative approach with a strategic artifact focus [30]. We provided an overview of notions of trace link correctness identified as applicable on the Data Model Level. For each notion, a general definition, information about generalizability and recommended experience level of the team was added. The general definition focused on explaining the main use and purpose of the notion. Generalizability detailed whether or not the notion is generic, and thereby applicable in the same form to any project, or if the application has to be customized for every project, i.e., it is project-specific. Finally, the recommended experience level defines whether or not the notion is recommended for use with a team composition consisting of software engineers which are not domain experts. The artifact was evaluated using semi-structured interviews in the first iteration and surveys in the second iteration. The research carried out an external evaluation as the evaluation was performed by people who are not involved in the development of the artifact.

A total of two iterations were conducted during this study. The study was conducted between February and May 2021. The first iteration took two and half months while the rest of the time was spent on the second iteration. The second iteration covered compiling the results, deriving conclusions, suggesting future work and communicating the research results.

3.2 Iteration 1

Iteration 1 focused on understanding the theoretical background of the problem, investigating the existing interpretation of notions of trace link correctness and creating the list of candidate notions of trace link correctness based on the gained knowledge.
The artifact was developed based on the findings in scientific articles and books. The initially developed artifact for this research resulted in a table where all potential notions of trace link correctness were listed. The table contained details for each notion of trace link correctness based on the context of their usage in the literature. The process of this iteration is depicted below in Figure 3.3.

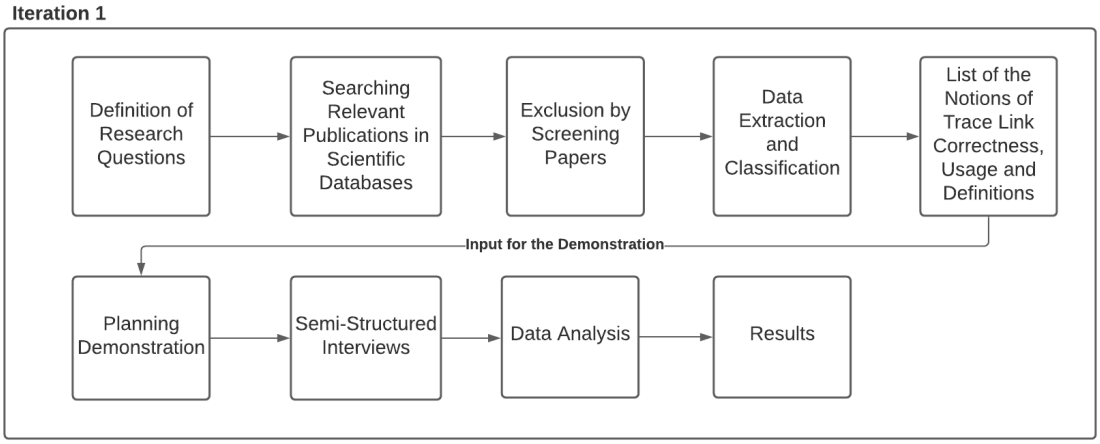

Figure 3.3: Summary of the research process of Iteration 1

3.2.1 Preparation

The Preparation phase describes how the artifact was developed and how the demonstration phase was prepared.

Initial Artifact Development

In this iteration, an exhaustive search in literature in the field of software traceability was performed. This was performed to get insights in the potential notions of trace link correctness that exist in the field.
We followed the guidelines for conducting a systematic mapping as suggested by [31], as shown in Figure 3.3 above.

Definition of Research Questions
See Section 1.

Searching Relevant Publications in Scientific Databases
Our goal was to find literature published on traceability and traceability links in

the domain of computer science. We used the Google Scholar search engine to find relevant publications. From there, we were forwarded to the ResearchGate, IEEEXplore, ScienceDirect, ACM Digital Library and CiteSeerX databases. We used search keywords related to traceability such as: software traceability, trace links, traceability correctness, traceability link mining, trace link maintenance, trace link creation, software traceability case study and traceability link quality.

Exclusion by Screening Papers
To select the papers relevant for our research the following inclusion criteria was set:

- The paper focuses on traceability links within computer science
- The paper includes information about trace link correctness
- The paper includes information about how trace links are generated or created
- The paper includes information about trace link validation
- The paper includes information about trace link evaluation
- The paper includes information about trace link maintenance
- The paper includes information about trace link quality
- The paper is written in English

For inclusion, the paper had to be written in English and also fulfil any of the inclusion criteria set. The Google Scholar search engine was used for searching for papers as it returns results from multiple scientific databases. Google Scholar limited the number of available publications to a maximum of 1000 publications per inclusion criteria. The number of papers for each inclusion criteria is presented in Table: 3.1 below:

Table 3.1: Total number of searched papers

Searching phrase	**Number of publications**	**Available for reading**
Software trace links	1 040 000	1000
Software trace link correctness	64 000	1000
Software trace link generation	402 000	1000
Software trace link creation	188 000	1000
Software trace link validation	165 000	1000
Software trace link evaluation	432 000	1000
Software trace link maintenance	188 000	1000
Software trace link quality	503 000	1000

With regards to the high number of search results and reserved time for conducting this phase of the research, the focus was on reading papers from the first 10 pages of the generated results in each category. The Title, Abstract and Introduction sections of the publication was prioritized to be read. This was done to identify papers which included information that was useful in this research. Additionally, keywords such as: “correct”, “trace link correctness”, “generated”, “maintain”, “trace link quality”, “quality”, “evaluate” and “validate” were searched for in each paper. When a keyword was found, the entire section where the word was mentioned was read to

check if the context of the searched word provided useful information with regards to notions of trace link correctness. It is important to note that in some cases when the content of a paper was found to be relevant to our research, the Bibliography in the research was checked which pointed to some other papers not found within 10 pages initially checked. This led to additional 12 discoveries used in this study.

A Google Sheets document was created to map all relevant information found related to the artifact. When a potential notion of trace link correctness was found, it was added to the sheet. When the same notion of trace link correctness was found in multiple sources, a maximum of three papers were added to the sheet. For some notions only a single reference was found while other notions were more common with multiple sources found.
This resulted in 26 relevant publications used in this research for development of the initial artifact. Each selected article was scrutinized for information about what approaches the researchers used in order to create, validate and maintain trace links. The focus was on identifying how the research had defined what a correct trace link was.

Data Extraction and Classification
In order to come to a definition of each notion, we grouped all articles describing the same notions of trace link correctness. Based on all articles and citations found for each notion of trace link correctness, we created a general definition of what that notion represents. It is important to state that even though several articles described the same notion of trace link correctness, the naming used in the articles varied between them in multiple cases. As an example "Directionality" and "Trace Link Direction" were two different names used for the same notion. Different names were initially recorded in the spreadsheet, but the notions were eventually merged to one name after the detailed analysis of each notion. In these cases, the most fitting of the names used in the articles was selected, based on the initial definition of the notion. Furthermore, for each notion of trace link correctness it was analyzed how the notion was used in each of the selected articles. The goal was to identify whether the notion of trace link correctness was applied during the creation, validation or maintenance of the trace link, i.e., the traceability Data Model Level, or if it was applied during the creation or maintenance of the traceability information model i.e. the Information Model Level. If all articles applied the notion on the same model level, the notion was classified as being useful in this level. If however, the articles applied the notion to both model levels, then the notion was classified as being useful in both level. As the purpose of this research was to find what defines a trace links as correct on the Data Model Level, all notions which proved to be applicable only on the Information Model Level were not analyzed in detail.

Preparation of Demonstration and Data Collection

Semi-structured interviews were used as a method to collect data to evaluate the created artifact. Semi-structured interviews were selected as data collection method

as they give control to the researchers to obtain information from interviewees, but they also give an opportunity to the interviewee to elaborate on a particular issue [32].

Creating the interview questions was the first step in preparing the interviews. The interview consisted of five main questions and depending on the answers some were followed by sub-questions. For the purpose of this research, we created a simple flow chart to visualize the flow of the interview process, see Appendix A.1. The interview flow covers three main aspects for the notion being examined: definition, application, generalization. The "definition" aspect provided knowledge on how the notion can be defined in practice, while the "application" aspect described how the notion of trace link correctness was applied in the project being analyzed by the interviewed domain expert. The "generalization" aspect clarified if the notion of trace link correctness could be applied on any project or if it was found to be project-specific.

The Convenience sampling method was used for the selection of the participant for this iteration [33]. Prior to the interview, all participants were provided with information about the objectives of the study, the purpose of the interview, terms used in the interview and the initial artifact via email. To avoid ethical implications of conducting the study, all interviewees were presented with a consent form asking if the interview can be recorded and assuring them that their anonymity would be preserved. The same information was repeated orally to the participants to eliminate any inconvenience regarding the interview or the research itself.

3.2.2 Demonstration

The artifact was evaluated on three different projects by conducting semi-structured [32] interviews with one domain expert in the context of each project. The first project used for the artifact evaluation was the iTrust dataset from the Healthcare Insurance Portability and Accountability Act (HIPAA). The dataset was selected from the Center of Excellence for Software and System Traceability (CoEST). The second project evaluated was the MobSTr dataset from the PANORAMA Research Project, while the third project was a project from an automotive company based in Sweden. As the content of the dataset from the automotive company was secret, we did not get allowance to publish the contents in this research. The second and the third projects were from same domain, i.e. the automotive industry domain. These projects were selected as we had access to relevant domain experts. Data was collected was in the form of audio recorded interviews.
To collect the data, semi-structured interviews were held with each domain expert separately. The duration of the interview was limited to 60 minutes. The interview flow included five main questions. Three out of five questions were supported by sub-questions. Sub-questions were designed in order to open the discussion when the main question was not sufficient to derive a clear conclusion. In total, three domain experts participated in the research. Each project was represented by one

domain expert each. One Quality Manager (HIPAA), one Functional Safety Engineer (MobSTr dataset) and one Project Manager (Automotive Company) were interviewed. Two interviews were conducted online where each interview was recorded in order to keep the interview flow fluent. We also took notes while conducting the interview to ensure none of the important information was missed. For this purpose Zoom, a tool for online meetings, was used to conduct the interview. Invitations for meetings were sent to the participants via email. One interview was held in person in a company office. This interview was recorded using a mobile phone. To ensure the privacy of the meeting, a separate meeting room was booked.

The interviews started with a presentation of the research followed by asking if the participant allowed that the interview was recorded. All participants agreed to this. After the recording started, we explained the terminology used in the interview to ensure that each participant understood the meaning of key words used in the context of the research. As the project datasets used in this research were fairly large, the researchers extracted a subset of the data to give a better overview and faster evaluation during the interview. This was visually represented in the form of UML diagrams for each project. The subset covered all permissible link types in the data model. Two domain experts were provided visual overview of the given trace link dataset specific for the project in their respective domain, see Appendices A.2 and A.3. One domain expert from the automotive industry presented a dataset used for the project in this domain. The dataset presented to us during the interview can not be attached in this research as we did not get approval to publish the dataset due to secrecy of the project used for the analysis.
The rest of the interview steps followed the flow represented in Appendix A.1. It is important to state that the flow of the interview slightly changed during the interview meeting due to time limitation. For the last four notions of trace link correctness we first asked about application level. If the notions were applicable on Information Level according to the domain experts, the detailed discussion about them was skipped. This did not affect the results of this research since the notions applicable only on Information Model Level were not the focus of the research.

3.2.3 Evaluation

The collected data was analyzed and interpreted using thematic analysis by employing a structural coding approach [34] to categorize themes based on interview questions. The data collected in this iteration provided us with feedback on the initially created artifact. The feedback was used for preparation of the next iteration and it provided us answers to RQ1 and RQ2.
As all interviews were recorded, the first step in this phase was to transcribe the data. We used a verbatim transcription approach [35] for transcription of interviews as some of the questions were not simple to answer but required additional elaboration. Making sure that each word from the interview was transcribed ensured that we do not miss the context of the answer. This simplified the analysis of the interview and resulted in increased correctness of interview interpretation. To transcribe

the data we used the oTranscribe[1] tool. Each interview was listened to once more by the researchers to ensure that the transcription of the data was correct. This resulted in a total of 44 pages of transcribed text.
The second step in the analysis was to create a template for the analysis in Google Sheets. Thematic approach [36] was used to identify themes in the data. Identified themes were organized by employing the structural coding [34] approach. Each notion of correctness was analyzed separately and each of them had the same structure for the analysis. The following themes were identified:

- Notion of Trace Link Correctness Definitions
- Application of Notion of Trace Link Correctness
- Applicable Model Level
- Generic or Project-Specific

Each theme had a sub-theme in context of each notion of correctness separately.

3.3 Iteration 2

Focus of this iteration was on understanding if a domain expert is required to evaluate trace link correctness using the identified notions. For this purpose, the artifact from Iteration 1 was updated according to the opinions of the domain experts whose feedback was classified as the ground truth for the artifact. The input for the data collection was a list of notions of trace link correctness applicable on the Data Model Level along with their definitions. For this iteration, the researchers selected to use the HIPAA project as it represents a simple and smaller data-set for evaluation. Questionnaires were sent out to 12 participants where six participants were students at the Software Engineering master's program at the University of Gothenburg or Chalmers University of Technology and six participants are software engineers from the industry with multiple years of working experience. None of the recruited participants had previous experience in a HIPAA project. Nominal data was collected and therefore the analysis was done by applying a grouping method [37] and the results for each group are graphically presented.
The process of this iteration is depicted below in Figure 3.4.

[1]https://otranscribe.com

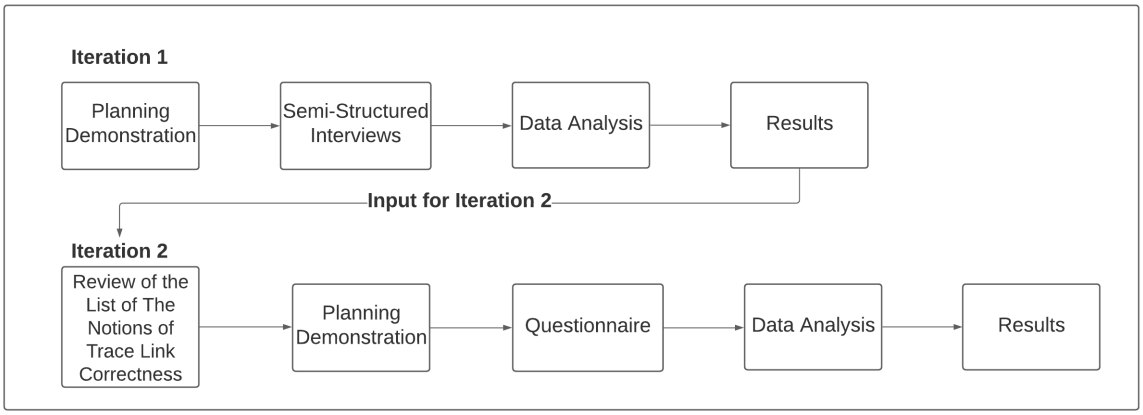

Figure 3.4: Summary of the research process of Iteration 2

3.3.1 Preparation

In this iteration, the preparation phase included the update of the initial artifact according to the results of Iteration 1. Furthermore, notions of trace link correctness that were not found to be applicable on the Data Model Level during Iteration 1 were discarded from the initial artifact. The updated artifact used as an input into Iteration 2 is presented in Table 4.7.

A questionnaire containing a combination of open-ended and close-ended questions was used as a method to collect the data. This type of questionnaire was selected as some of the questions required additional information in order to derive the conclusions correctly. The questionnaire consisted of two main questions followed by additional sub-questions which required additional elaboration in free-text form from the respondent. The questionnaire for the data collection was based on the main question flow for the interviews in Iteration 1, but questions not relevant to the purpose of this iteration were removed, see Appendix B.1. The questionnaire was included in a document containing the description and purpose of the research, list of notions of trace link correctness and their definitions and a visual representation of the iTrust/HIPAA data-set shown in A.2. Additionally, a consent form informing participants about participation in the research, benefits, risks and confidentiality of their personal information was also provided.

3.3.2 Demonstration

The test subjects were divided into two groups: experienced software engineers, represented by participants with multiple years of working experience, and non-experienced software engineers represented by Software Engineering students at the University of Gothenburg and the Chalmers University of Technology. The questionnaire was sent to each participant separately via e-mail. The deadline for filling out the questionnaire was eight days after they received the e-mail. 12 out of 15 contacted test subjects responded which resulted in an 80% response rate.

3.3.3 Evaluation

The evaluation of the data collected was done using Google Sheets where all responses were mapped respectively to each of the participants. From the mapped responses, a frequency distribution table was created from which graphs were derived. Three graphs were created for each notion of trace link correctness. The categories analyzed were: Definition, Data Exists and Data to Extend. The *Definition* category provided insights if the participants understood the general definition of each notion. The *Data Exists* category explained if the participants understood the given project's dataset and if they could recognize patterns of each notion existing in that dataset. *Data Extend* showed if the participants were able to apply each notion on the given dataset. For the evaluation of responses, each response was compared to the stated opinion of the domain expert from Iteration 1, i.e., the ground truth of the study, by the researchers. If the response was the same, it was classified as being in agreement with the domain expert. If the response differed, it was classified as being in disagreement with the domain expert. Learnings from this iteration were added to the artifact.

3.4 Threats to Validity

Validity indicates the degree of reliability of the study. This is vital for denoting unbiased results from a researcher standpoint [38]. In this section we discuss the main threats to validity of the study and the measures taken to minimize these threats. The categories of validity proposed by Runesson and Höst were adopted and explain in the sections below.

3.4.1 Internal Validity

Internal validity concerns the trustworthiness of the relationship between a treatment and the outcome that it provides, for instance cause-and-effect. Therefore errors or inconsistencies in the underlying studies of the papers used to build the artifact can potentially impact the result of this research. However, to minimize this threat the papers used to build the artifact were primarily found in the Google Scholar search engine. All the referred papers were extracted from scientific journals and books for software engineering.

Another threat is a potential error in the search process. This kind of limitation is particularly difficult to tackle given that errors for omission can occur and research papers and books containing notions of trace link correctness could potentially be overlooked. To address this threat, the papers used for the systematic review of papers focused on researches about *software traceability, trace links, traceability correctness, traceability link mining, trace link maintenance, trace link creation, software traceability case study and traceability link quality.* Moreover, the papers have been selected by using a set of inclusion criteria described in Section 3.2.1, reducing the problem to some extent.

As the test subjects for Iteration 2 were not domain experts and had no previous knowledge regarding the details of the project, there was a risk that the participants did not understand the given dataset well enough to conclude whether the notions of correctness were applicable. To reduce the risk to some extent, some of the test subjects recruited were from the industry and have years of practical experience with software traceability.

Issues related to instrumentation can arise during the data collection and analysis phase. According to Wohlin et al. [39], the way in which the data collection is designed can potentially affect the results of the analysis. In order to minimise this threat, the material, such as interview and questionnaires survey questions, visualisation models as well as the general guidelines and procedures, were shown and discussed with the academic supervisor prior to their application.
A pilot questionnaire was sent to two participants to ensure that it was clear and understandable. As the feedback was positive, the pilot questionnaire remained unchanged when it was sent to the other participants and the answers from these two participants was therefore included in the results. To mitigate the biases in the analysis of the data collected during the interviews with domain experts, the researchers followed the guidelines proposed by Runeson et al. [38]. The interview sessions were divided into different phases: Presentation of the study, handling and use of data, introductory questions about the participant and the interview questions.

Given that the study was limited to nineteen weeks, the time-frame allowed for the inclusion of only three projects and the process for each iteration was compressed accordingly to fit the corresponding scope. The first Iteration had a duration of ten weeks and the second nine weeks. Since the period of the study was limited the amount of data that could be collected and analysed was also limited. For example, the number of projects and participants could have been increased if the project time frame allowed. However, the projects used to verify the developed artifact belong to two different domains. By doing so we ensure the heterogeneity of the results. To mitigate this threat as best as possible given the low number of projects, two projects were from the same domain and the third project was from a different domain. This way, there was a representation of multiple projects within one domain. The results are extracted from two very different domains.

3.4.2 External Validity

This threat refers to the extent to which the results of a study can be generalized. Data triangulation was applied to ensure the validity of the results [40]. This was done by using three different projects and performing the survey and interviews with participants involved in different roles and expertise within software engineering but none of them had any previous knowledge with the analyzed project domain. We increased the probability to capture different dimensions of the study while at the same time obtaining data from different sources. We recruited an heterogeneous group of 12 software engineers and students within software engineering that work

in different roles, have different working experience and are active in different industrial areas such as telecom, governmental agency services as well as the automotive industry.

Regarding the systematic review of papers, there exists the possibility that papers containing additional notions of correctness have been published while and after the artifact was build. This can potentially lead to erroneous conclusions and an incomplete artifact. However, this is a limitation we accept and acknowledge.

In order to ensure the reliability and validity of a study, it is imperative that this is reproducible by other researchers and achieve similar results. However, the setting in which the interviews and surveys have been constructed cannot be replicated. Nonetheless, the definitions, visualisation models as well as the process followed during the interviews were well documented and can be used to replicate the study. The main objective was to comprehend the different perceptions to form unbiased insights of the conducted research.

An attempt to generalize the results was made. Nevertheless, there exists a need to analyze more projects and apply the different notions in order to determine the degree of generalizability. This study was carried out in very specific settings. The authors used methods and processes described in this paper. In addition all steps have been documented.

3.4.3 Construct Validity

Due to the fact that the researchers of this paper are not experienced in conducting this type of study, a risk of construct validity arises. The visualisation models presented and used as input for the interviews and surveys could potentially contain errors and be misinterpreted. This leads to potential inconsistencies and unreliability of the study. In order to mitigate this risk, the data used for the study was extracted from the Center of Excellence for Software and System Traceability (CoEST), the PANORAMA Research Project for automotive and aircraft industry taken from GitHub (https://github.com/panorama-research/mobstr-dataset) and from projects in a well-known Swedish automotive company.

As Wohlin [39] states, misinterpretation is a risk that involves all the different parties in the study. Hence, another risk is that during the interviews and surveys the instructions and material presented to the participants were misunderstood or misinterpreted. To reduce the risk, the material was sent prior to the interview. During the interview the information was given once more orally. Moreover, the semi-structured interview strategy allowed the participants to ask questions when needed.

To ensure comprehension, the researchers used semi-structured interviews with open-ended and close-ended questions. This encouraged discussions and clarifications between the participants and researchers. Additionally, we ensure that the general idea

of the study was understood by describing the concepts of the study at the beginning of the interviews. The questionnaires sent to the participants contained detailed information about the study. To ensure the correct interpretation of the questions, graphs and an example were utilized, see Appendix A.3. Moreover the purpose of the study and the procedures for data handling were also clarified, see Appendix B.1.

3.5 Ethical Consideration

Regarding confidentiality, Coffelt [41] denotes the importance of confidentiality and the role and responsibilities of the researchers. For example, they are responsible for protecting the identities of each participant ensuring that the information provided by the participants is properly managed and only disclose relevant information to the study. All participants of the surveys and interviews were informed about the confidentiality and anonymity through a consent form that was presented and explained prior to the data collection. The form had the following points: (1) Procedures: The participant is aware of the procedures and agreed to participate in the study. (2) Ensuring Confidentiality: The participants acknowledge that the data collected will remain confidential and anonymous. (3) Permission to Record the Interview: The participant is aware that the interview will be recorded and transcribed. (4) Permission to Quote: The participant is aware and agrees that, if needed, their direct quotes from the interviews might be used anonymously in the discussion.

4
Results

This section provides results from the data analysis. Deeper analysis was performed on notions found to be applicable to the Data Model Level as this was the main focus of the research.

4.1 Iteration 1

The purpose of this iteration focused on answering *RQ1: What notions define trace link correctness?* and *RQ2: Are the defined notions of trace links correctness generic or project-specific?*
The analysis of the data collected was divided into five themes following the structure of the interview questions. Each theme has a sub-theme in context of each notion of trace link correctness separately.

4.1.1 Initial Artifact Design

In the following, we present the different notions of correctness as we derived them from the literature. The initial definition of each notion was written in the form of a guideline. This was done to better explain the concept of each notion to the domain experts, thereby supporting the main purpose of Iteration 1, i.e., evaluating the discovered notions of trace link correctness. Details about how we developed this initial version of the artifact can be found in Section 3.2.1.

- **Versioning**
  A type of versioning was found in the work by Gall et al. where they stated that historical changes of classes need focus [42]. A log must be created containing information about when new artifacts are added and when existing ones are modified.

  "Put focus on historical changes of classes. The time when new classes are added to the system and when existing classes are changed has to be measured." [42]

  Maro et al. stated that if a versioning solution exists, then the traceability model must be updated with respect to the new versions in the model [3]. The research also included judging whether a trace link should remain between

the updated artifacts or not. This implies that Versioning is a notion that is applicable on the Data Model Level.

"Maintainer must check if there are implications caused by evolving connected models. If a versioning solution exists, the traceability model must be appropriately updated with respect to the new versions of the models, i.e., one must decide if there should (still) be a link or not." [3]

A summary of the findings is that if Versioning is used in a project on the Data Model Level, a trace link is to be considered correct if it links to the latest version of each linked artifact. If a later version of a linked artifact exists, the trace link must be reviewed for decision if it is to be updated or removed. A change history log should exist and it supports the review process by simplifying understanding of updates performed.
The initial definition of Versioning as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to the Data Model Level.

- **Lifetime/Lifespan**
  Lifetime, as a notion of trace link correctness, was found in the work by Maro et al. [3]. They stated that during the lifetime of a project, link types can be added and old ones can be removed. As the work specifically mentions link types, it refers to the Information Model Level.

  "The semantics must thus be adapted and updated continually during the lifetime of the project, not only by adding new "types" of links, but also by refining and even deactivating existing types." [3]

  Contrary to this, another type of Lifetime was found in the work by Cleland-Huang et al. where they stated the need to differentiate between throw-away and long-term trace links. These trace links are found in the Data Model Level and the suggestion focuses on how long the trace links are to be maintained.

  "Differentiate between throw-away and long-term traces. Throw-away traces that are useful only during development, and those that should be maintained in the long-term." [43]

  A summary of the findings is that if Lifetime/Lifespan is used in a project on the Data Model Level, a trace link is to be considered correct if it's maintenance expiration date is not due. On the Information Model Level, Lifetime/Lifespan cannot be used for evaluation of correctness, but rather mandates that regular review is performed on the project TIM to secure that the used link types fit the current needs of the project.
  The initial definition of Lifetime/Lifespan as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to both levels as the two statements cover usage of the notion in one level each.

- **Non-Duplicated Trace Links**
  A clear definition of Non-Duplicated-Trace Links as a notion of trace link correctness was found in the work by Mäder et al. [14]. It was suggested that the solution is oriented towards using a database-like repository which automatically detects duplicated links and removes them. As a database was mentioned for storing created trace links, the description fits the Data Model Level.

  "Prevent duplicated links by storing them in a database-like repository. Either define constraints that prevent redundant links from being created or regularly execute trace queries to find duplicated links and remove them." [14]

  A summary of the findings is that if Non-Duplicated Trace Links is used in a project on the Data Model Level, a trace link is to be considered correct if there are no duplicates of the trace link.
  The initial definition of Non-Duplicated-Trace Links as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to the Data Model Level.

- **Unique Artifact Identification**
  Unique Artifact Identification was a notion of trace link correctness identified in the work by Mäder et al. [14]. It referred to having unique identifiers to each traceable artifact. As traceable artifacts exist only in the Data Model Level, the work was found to suggest the Data Model Level as the correct level where this notion can be used. There was also a recommendation in their work on using fixed prefixes for all artifact classes found in the information model as part of the artifact identifier in the data model. This suggested that unique prefixes must be added to each artifact in the Information Model Level, i.e. the notion is applicable to this level as well.

  "A fundamental principle of traceability is that each traceable artifact must have a unique identifier. Furthermore, prefixes used to distinguish artifact types should be unique across the project as well as intuitive to stakeholders." [14]

  A similar definition of Unique Artifact Identification was found in the work by Maro et al. [44]

  "Artifacts need to be unique which is a characteristic of a traceable artifact." [44]

  A summary of the findings is that if Unique Artifact Identification is used in a project on the Data Model Level, a trace link is to be considered correct if all artifacts it links to can be uniquely identified using their ID. On the Information Model Level, the TIM is to be considered correct if all artifact

classes can be identified by a unique prefix.
The initial definition of Unique Artifact Identification as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to both levels. In the initial definition of the notion, only the part applicable on the Data Model Level was used as the research focused only on this level.

- **Mandatory Artifacts & Mandatory Trace Links**
  Rempel et al. suggested in their work that guidelines may exist within a project which requires some specific artifact types to be created within a project [21]. This would have an impact to both the Information Model Level, as it would have to contain these artifacts, and the Data Model Level, where the required artifacts and required traces between them could be instiated.

  "The artifact types required by a guideline must be available within a project in order for traceability to be established between them." [21]

  A similar suggestion was found for mandatory trace links, where the Information Model Level must contain permissible link types in order to allow for the mandatory trace links to be created in the data model of the project at some stage.

  "Traceability required by a guideline between artifact types can only be established at the project level if a trace path between the two artifact types is available at the project level." [21]

  This was also elaborated further in the work where it was clarified that either trace links or trace paths can be setup as mandatory in a project.

  "Traceability between artifact types that is required by a guideline can only be considered complete if at the project level every single artifact of the requested source artifact type is traced directly via a tracelink or transitively via a trace path to an artifact of the requested target artifact type." [21]

  A summary of the findings is that if Mandatory Artifacts & Mandatory Trace Links is used in a project on the Data Model Level, a trace link is to be considered correct if it links to required artifact types in accordance to the project guidelines. On the Information Model Level, the TIM is to be considered correct if all artifact classes required by guidelines are defined. Additionally, each trace path and permissible trace link required by the guidelines is defined in the TIM.
  The initial definition of Mandatory Artifact & Mandatory Trace Links as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to both levels.

- **Link Type**
  Agrawal et al. grouped various types of relations and recommended what link types to use in a traceability model [45] when constructing the project TIM.

  "In this paper, we organise the various types of traceability relations proposed in the literature into eight main groups namely: dependency, generalisation/refinement, evolution,satisfaction, overlap, conflicting, rationalisation, and contribution relations." [45]

  In similar fashion, Zisman et al. stated that link types should be defined in a way which suits the intended use, i.e., what type of tracing will be performed [5]. This also focused on setting up permissible link types in the Information Model Level according to well defined goals.

  "Link types should be defined in a way suitable for the intended usage. In the following we list a set of activities in which the links are used, together with the goals of using them.
  - Verification of (forward) engineering activities
  - Change impact analysis
  - Software comprehension and reverse engineering
  - Identification of the source of a decision or requirement
  - Decision support
  - System configuration and versioning" [5]

  A summary of the findings is that if Link Type is used in a project on the Information Model Level, the TIM is to be considered correct if each permitted link type defined clearly describes the relation between the linked artifact classes.
  The initial definition of Link Type as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to the Information Model Level.

- **Granularity**
  Granularity as a definition of trace link correctness was found in the work of Mäder et al., where it was stated that it is important to clearly define the granularity of permissible trace links in the TIM [14]. This suggestion thereby stated that the notion is applicable on the Information Model Level. Furthermore, it was recommended that on the Data Model Level, regular evaluations are performed in order to verify that all created trace links follow the granularity level found in the project TIM. This also implied that the notion is not a notion of trace link correctness on the Data Model Level as the only requirement was that the trace links were created to the same granularity as that defined in the TIM, i.e., follow the project TIM in general.

  "Trace link granularity must be clearly defined, in the TIM and RTMs and

must be periodically evaluated in order to ensure that trace links are created at the correct granularity." [14]

The same statements were found in the work by Ghabi et al where it was suggested that the TIM should explicitly define the correct granularity level for every permissible trace link [17]. Again, it was recommended that created trace links should be checked regularly to ensure that they follow the TIM.

"The solution suggested is that the granularity of the links should be defined explicitly in the traceability metamodel and the traceability links should be checked regularly to ensure that the links are created with the right level of granularity." [17]

The notion was also identified in the work by Zisman et al. [5], stating that software specifications can exist on different levels of granularity.

"As part of rationalization, relations are expressed between traceable specification, a software specification with different level of granularity such as document, model, diagram, use case, etc." [5]

Javed et al. stated that automated approaches have difficulties working with trace links on various levels of granularity [27]. This provided more evidence that the notion is important if automated tools are to be used for maintenance of traceability data in a project.

"Automated approaches tend to have difficulties working with various levels of granularity." [27]

A basic definition of what Granularity is, was found in the work by Paech et al. [26]

"The granularity describes the granularity of the entities involved (e.g., classes or attributes/methods of an object-oriented analysis, paragraphs or sentences of a textual requirements document)." [26]

A summary of the findings is that if Granularity is used in a project on the Information Model Level, the TIM is to be considered correct if the granularity level for all permissible link types is defined.
The initial definition of Granularity as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to the Information Model Level. It was not classified as belonging to the Data Model Level as none of the statements clearly defined how it would be used on this level other than following what was defined in the TIM of a project.

- **Purpose**

Purpose was a notion of trace link correctness found in the work by Cleland-Huang et al., where it was used to ensure that each trace link has a reason for existing [43]. It was stated that the purpose will aid in the selection of the most useful link types to be deployed on the Information Model Level.

"In order to minimize negative-return traces, it is important to evaluate why a link is being created, so that the most appropriate and useful type of link can be deployed." [43]

A summary of the findings is that if Purpose is used in a project on the Information Model Level, the TIM is to be considered correct if the purpose of each permissible trace link is defined.
The initial definition of Purpose as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to the Information Model Level.

- **Non-Redundancy of Traceability Paths**
  Mäder et al. found that redundant traceability paths defined in a TIM could lead to issues in maintaining and using the project traceability matrices [14]. The focus was on avoiding more than one path from one artifact to another in the TIM.

  "Redundant traceability paths defined in the TIM lead to extraneous and possibly diverging traceability matrices. A TIM includes a redundant path if there's more than one way to trace from one artifact type to another." [14]

  A summary of the findings is that if Non-Redundancy of Traceability Paths is used in a project on the Information Model Level, the TIM is to be considered correct if only a single path exists from any artifact class to all other artifact classes.
  The initial definition of Non-Redundancy of Traceability Paths as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to the Information Model Level as the suggestion is to evaluate it on the project TIM.

- **Arity**
  Maletic et al. identify Arity as a notion of trace link correctness [46]. The notion specified the number of end points on a trace link. This suggested that the notion of trace link correctness is to be used on the Information Model Level as this is where the definition of permissible trace links is defined in a project.

  "The arity of a link specifies the number of its end points." [46]

A summary of the findings is that if Arity is used in a project on the Information Model Level, the TIM is to be considered correct if the number of end points of each permissible trace link is defined.
The initial definition of Arity as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to the Information Model Level.

- **Directionality**
  Directionality was found as a notion of trace link correctness in the work by Javed et al., where the need for unidirectional trace links and bidirectional trace links was found [27]. Examples of when unidirectional trace links could be used was also provided.

  "Unidirectional traceability approaches supports to establish 'trace to' links from one artefact to another (also called forward traceability). Examples where these type of trace links have been used are links from code base to architecture and from code base to architectural tactics. Bidirectional traceability approaches support forward as well as backward traceability." [27]

  Maletic et al. also suggested directionality as a notion of trace link correctness, but the possible definition was somewhat extended. Navigational directionality was defined very similarly to what was defined as directionality in [27], but another layer was added as trace links could have a Logical directionality.

  "Link directionality takes two forms: navigational and logical. Navigational directionality refers to the direction(s) in which a link may be traversed. Logical directionality is a semantic quality that is independent of how a link can be traversed." [46]

  A summary of the findings is that if Directionality is used in a project on the Information Model Level, the TIM is to be considered correct if each permissible trace link is defined as uni-directional or bi-directional. Additionally, the direction of all uni-directional trace links is defined.
  The initial definition of Directionality as a notion of trace link correctness can be seen in Table 4.1 and it was classified as belonging to the Information Model Level as both references focused on defining the permissible trace links and their direction on this level.

Table 4.1: Initial List of Notions of Trace Link Correctness and Their Definitions

Notion	Initial Definition	Model Level
Versioning	Versioning is about maintaining a change history log and securing that information about the evolution of each artifact is stored. As part of the concept, each change to any given artifact can be logged and this enables tracing changes to a certain point in time.	Data Model Level
Lifetime/Lifespan	Lifetime/Lifespan is about setting an end date to each trace link where maintenance will no longer be performed.	Information Model Level/ Data Model Level
Non-Duplicated Trace Links	Non-Duplicated Trace Links is about securing that there is no duplicated information in trace link data. This in order to avoid future update errors where only one link has been updated but not other duplicates. This makes it impossible to know which one is correct and which link was not updated correctly at some point in the project life cycle.	Data Model Level
Unique Artifact Identification	Unique ID is about being able to identify a specific artifact through it's ID. The ID only belongs to one single artifact in the project, thereby identifying it uniquely via the ID.	Information Model Level/ Data Model Level
Mandatory Artifact & Mandatory Trace Links	Mandatory Artifact is about securing that all artifacts that must exist according to used guidelines or regulations are created. This secures that applicable guidelines and regulation are fulfilled by the project. Mandatory Trace Links is about securing that all trace links that must exist according to used guidelines or regulation are created. This secures that applicable guidelines and regulation are fulfilled by the project.	Information Model Level/ Data Model Level
Link Type	Link Type is about defining the utilization and goal of a trace link. Example: implementedBy, testedBy	Information Model Level
Granularity	Granularity is about relating a link to a specific level of an artifact. Each artifact can have various levels. Examples could be file, class, method and line of code as different levels for the same code artifact. Another example could be requirements document, specific requirement and specific step in a specific requirement as different levels when linking to requirement artifacts.	Information Model Level
Purpose	Purpose is about securing that each permitted trace link serves a specific purpose and thereby provides value. This means the effort to create and maintain the trace links brings return on invested resources.	Information Model Level
Non-Redundancy of Traceability Paths	Non-Redundancy of Traceability Path is about avoiding multiple paths from one artifact to another via trace links. Having multiple paths makes it more difficult to maintain the traceability model and the risk of introducing errors is increased.	Information Model Level
Arity	The arity of a link specifies the number of its end points. For example: one-to-one, many-to-many, one-to-many.	Information Model Level
Directionality	Directionality is about defining the tracing direction of a trace link. Trace links can be unidirectional, i.e. implement forward traceability, and bi-directional, i.e. implement both forward and backward traceability.	Information Model Level

4.1.2 Applicable Model Level

The results in this section provided us with an answer to RQ1: *What notions define trace link correctness?* For this purpose the goal was to discover whether the domain experts participating in this research agreed with the literature. Here, we investigate to which model level each notion of trace link correctness applies. The data collected and justification of the final decision is described for each notion of trace link correctness below and an overview of the results is presented in Table 4.2 below.

- **Versioning**

Two domain experts agreed with the findings in the literature that Versioning is applicable on the Data Model Level.

"I think the information module is more static and more defined in that case, it wouldn't benefit as much from version control" (Interviewee 2)

"Versioning would be applied to the produced artifacts and not the artifact types in the information model" (Interviewee 3)

However, a conflicting opinion was found from one of the domain expert who would rather have Versioning applied on the Information Model Level. The main motivation for this opinion was that it would get very tedious to have Versioning in the Data Model Level. It adds a lot of information and data.

"I think if you go on a data model, if you would apply Versioning on a data model, it would get very tedious because it's a lot of information and a lot of data, I would personally prefer to have it on the information model" (Interviewee 1)

The domain expert used Versioning of hazard analysis as an example and their connection to hazards and requirements, stating that when an update has been performed on the hazard analysis then the version number should be updated in order to indicate that modifications have been made to this artifact. This is a weak motivation for using the notion on the Information Model Level as the hazard analysis should be an instantiated artifact in the Data Model Level.

"If you update the hazard analysis, you should increase the version number" (Interviewee 1)

The final decision during the research was to classify Versioning as applicable to the Data Model Level as this was the opinion of the majority.

- **Lifetime/Lifespan**
  A good motivation on why to use Lifetime/Lifespan as a notion of correctness was provided by one of the interviewees:

  "I'm a little bit afraid if you don't clean up data after a while, data might be invalid or not usable anymore. Then you end up with a lot of rubbish data that no one knows what to do with it." (Interviewee 1)

  A good explanation on how Lifetime/Lifespan can be used on both model levels was provided by one of the participants. The notion can be used on the Information Model Level to define how long each artifact type should exist and be maintained. On the Data Model Level, each artifact is then tagged with a precise end date. This is achieved by a standard within the company named

"The Global Records Standard" and it is used for all projects by the company the domain expert works for. The standard defines the retention time for each artifact type existing in the company. The standard does not seem to include a retention policy for trace links and a good reason for this was not provided.

"In the information model you define how long different types of artifacts must be stored from the date they are created or from the date the project went into production. The end date itself for each artifact must be specified in the Data Model as that date will vary per project it belongs to." (Interviewee 3)

"The lifetime in the form of number of years after the start of production is defined in the Global Records Standard." (Interviewee 3)

Another domain expert was of a similar opinion, but was not as sure on how it could be applied on the Information Model Level. However, it was no conflicting opinion as the domain expert clearly stated that the notion is applicable on both levels.

"First of all, of course, on the Data Model, I'm just thinking if it's also on the information model. But in theory, yes, it applies to the information model as well." (Interviewee 2)

The third participant showed concern on applying the notion on the Data Model Level. The domain expert's main concern was that it might get too complicated to use the notion on the Data Model Level. It was not a question of whether or not it can be used, but whether or not it is wise to use it on this level.

"It gets very complicated if you put it on the data level. So I think it should be on the information level" (Interviewee 1)

"I can't really say yes or no whether it's wise to have data of lifespan on data, but I'm definitely sure it's wise to have lifespan on information models" (Interviewee 1)

The final decision during the research was to classify Lifetime/Lifespan as applicable to both the Information and Data Model Levels.

- **Non-Duplicated Trace Links**
  Two domain experts share the same opinion as the literature. Both domain experts agreed that the notion makes most sense to apply on the Data Model Level only.

  "Data model would mean that you might have links, that we have one piece of data and several links that connect this piece of data to the same artifact.

That will not make so much sense, would it?" (Interviewee 1)

"Data Model for the implementation as this is where you would actually check for duplication of trace links between the same artifacts". (Interviewee 3)

However, the third domain experts was of the opinion that the notion can be used on both levels and for very complicated models the notion would be easier to use on the Information Model Level:

"It holds true for both. Much easier, of course, on the information model". At least if we're looking at very complicated information models. (Interviewee 2)

This indicates that Non-Duplicated Trace Links could be applicable on both levels, but the domain expert is of the opinion that it would be easier to apply on the Information Model Level. No further detail was provided by the domain expert on how the notion could be applied on the Information Model Level and what this would actually mean. Duplication of trace links on the Information Model Level should most likely be duplication of traceability paths, which is identified as a separate notion in this study. As this research focused on the Data Model Level, no further inquires were made to understand how the notion could be used on this level. The final decision during the research was to classify Non-duplicated Trace Links as applicable to the Data Model Level as both the literature and all domain experts had agreed that it could be used on this level.

- **Unique Artifact Identification**
  Two of the interviewed domain experts agreed that Unique Artifact Identification is a notion of trace link correctness which is applicable on the Data Model Level.

  "The data model, because every piece of data could have a unique ID and then it's differentiated." (Interviewee 1)

  "Data Model as this is where we would apply the unique IDs which differentiated them all". (Interviewee 3)

  The third participating domain expert was of an opinion aligning more with findings from the literature, which was that the notion is applicable on both levels. However, the domain expert also expressed that the notion is of most relevance in the Data Model Level.

  "For sure in both. It's most relevant for the data model". (Interviewee 2)

  No further details were provided on how the notion could be used on the

Information Level and as the study was focused on the Data Model Level, no elaboration was asked for during the interview. It could be an indication that the notion could be used in a similar fashion as described by Mäder et al. where each artifact class of the TIM is assigned a unique identifier [14]. This identifier is then attached as a prefix to the unique identifier of corresponding instantiating artifact in the Data Model Level. The final decision during the research was to classify Unique Artifact Identification as applicable on the Data Model Level.

- **Mandatory Artifact & Mandatory Trace Links**
  Two domain experts aligned with the literature opinion that the notion is applicable on both the Information Model Level and the Data Model Level.

  "Both really. You would have to define which artifacts and trace link types are mandatory in the information model so you know which ones you must have and their relation to other artifacts in your project. But then of course, you really need to have artifacts created in the Data Model Level which match those specified in the Information Model as mandatory. So I guess my answer here is both really". (Interviewee 3)

  One of the domain experts stated that this notion of trace link correctness is applicable on the Data Model Level with the argument that the data model should fully cover the information model.

  "It's definitely on the data model because you cover everything in the information model". (Interviewee 2)

  As the study focused on the Data Model Level, the domain expert was not asked to further elaborate the answer. The final decision during the research was to classify Mandatory Artifact & Mandatory Trace Links as applicable to both the Information and Data Model Levels as this was the opinion of the majority.

- **Link Type**
  One of the domain experts preferred to apply Link Types on both Model Levels.

  "I would have it on both, there might be data that is used by a certain artifact, and that information is actually quite important, especially now, when you have building more and more with data". (Interviewee 1)

  As a motivation for this need the training of AI systems and the need to identify which data sets have been used to reach the current system behavior is used.

> *"If you train an artificial intelligence system, a system that contains some form of artificial intelligence, the behavior of that system is defined by the data that you used to train it. So there is a clear link between the data set that you use for training and the behavior of the system. If you do not have this link type or state in the link type that this data is used, such as, "this AI model uses this training data and is tested by this other data set", then you are kind of losing the overview of how the behavior in the system has been created".* (Interviewee 1)
>
> The final decision during the research was to classify Link Type as applicable to the Information Model Level as this was the opinion of the majority. It would be interesting to further study the examples given by the domain expert of different opinion to better understand if this notion of correctness might be a useful notion to include on the Information Model Level for future AI development related projects.

The notions of granularity, purpose, non-redundancy of traceability paths, arity and directionality were all found to be part of the information model level according to literature and all interviewed domain experts. Therefore these notions are identified as belonging to the Information Model Level and were not further analysed.
In the table below IML stands for Information Model Level and DML stands for Data Model Level, while DE stands for domain expert.

Table 4.2: Overview of data results for applicable model level of Notions of Trace Link Correctness

Notion	Literature	DE1	DE2	DE3	Final Result
Versioning	DML	IML	DML	DML	DML
Lifetime/Lifespan	DML/IML	DML/IML	DML/IML	DML/IML	DML/IML
Non-Duplicated Trace Links	DML	DML	DML/IML	DML	DML
Unique Artifact Identification	DML/IML	DML	DML/IML	DML	DML
Mandatory Artifact & Mandatory Trace Links	DML/IML	DML/IML	DML	DML/IML	DML/IML
Link Type	IML	DML/IML	IML	IML	IML
Granularity	IML	IML	IML	IML	IML
Purpose	IML	IML	IML	IML	IML
Non-redundancy of Traceability Paths	IML	IML	IML	IML	IML
Arity	IML	IML	IML	IML	IML
Directionality	IML	IML	IML	IML	IML

4.1.3 Definitions of Notions of Trace Link Correctness

The results in this section were used to update the definition for each notion of trace link correctness from Section 4.1.1 according to the domain experts knowledge and experience. The data analyzed in this section was extracted from the interviews where the participants were presented with definitions of trace links notions created based on literature. The main goal was to asses whether the domain experts agree with the initial definition of the notions of trace link correctness. An overview summarising whether or not the domain experts agree with the initial definitions from Section 4.1.1 is shown in Table 4.3 below. The updated definitions of each notion of trace link correctness are presented in Table 4.7 below. These updated definitions were later used in Iteration 2.

- **Versioning**
  One out of three domain experts did not fully agree with the initial definition of Versioning. The main aspect missing from the definition was that Versioning as a notion can be used to synchronize the development effort of multiple development teams working on separate artifacts across a project.

  "I missed one aspect that I think Versioning can do. And that is the aspect of synchronizing different artifacts. We definitely also use it as a tool to keep different developments in sync." (Interviewee 1)

- **Lifetime/Lifespan**
  One of the domain experts considered that the definition should be extended to cover obligations and not only maintenance. In some products the developers might decide that no further maintenance will be done to the software, but you still have to take care of it as long as it is used due to legislation.

  "You could have a project or you could have a product, where you say from this date, we are not going to maintain it anymore. But if it's for example, a medical device, you still have obligations when it comes to post market surveillance. If something happens where this product is used." (Interviewee 2)

- **Non-Duplicated Trace Links**
  For this notion one of the domain experts considered that there should be an addition to the definition. The expert stated that if there is a need to actually allow duplicated trace links, an update to one of the links should also automatically update the duplicated trace links. In some cases it might be beneficial for the project to have duplication in the form of clones and a definition allowing and handling this should be created.

  "Maybe sometimes it is wise to have a duplicate of a trace link. But a link between the two links saying this link is a clone of the other link should exist.

And if you're updating one of the two, it should automatically trigger an update of the other one as well." (Interviewee 1)

- **Unique Artifact Identification**
  All domain experts agreed with the initially created definition.

- **Mandatory Artifact & Mandatory Trace Link**
  One of the domain experts used Functional Safety and Cyber Security as an example of why the definition requires an update. For these type of projects there are often regulations which require the creation of certain artifacts and trace links during product development.

  "There might be regulations that require that certain products or certain artifacts during your system development are created. For example, a safety case of your system for Functional Safety. The safety case needs input. You cannot just magically write a safety case, you'll need a hazard analysis, for example, you need a system analysis, you need analysis of the probability of random half reports, all these kinds of things, they need to be there, and they need to have a link". (Interviewee 1)

  The main modification was that mandatory trace links should not expire. If the artifacts and trace links are mandatory, they should most likely remain throughout the project development. After the product has launched the mandatory artifacts and trace links should be stored as proof of design decisions taken during the development phase of the project.

  "These trace links that are mandatory should also not expire, because if they expire, then they cannot be mandatory". (Interviewee 1)

Table 4.3: Overview whether or not the domain expert agreed with the initial definition

Notion	DE1	DE2	DE3
Versioning	No, the definition should be extended	Yes	Yes
Lifetime/Lifespan	Yes	No, the definition should be extended	Yes
Non-Duplicated Trace Links	No, the definition should be extended	Yes	Yes
Unique Artifact Identification	Yes	Yes	Yes
Mandatory Artifact & Mandatory Trace Links	No, the definition should be extended	Yes	Yes

4.1.4 Application of Notion of Trace Link Correctness

The results of this theme focused on partially answering RQ2: *Are the defined notions of trace links correctness generic or project-specific?*. The goal was to identify what data should be included in a project dataset to support evaluation of trace links using the identified notions of trace link correctness. The first step was to investigate whether or not data supporting the notion being analyzed already existed in the current project dataset. If no supporting data existed, each domain expert was asked what data should be added. If the results showed that all projects required the same data to support the notion, a strong indication had been found that the application of the notion was generic. However, if different data was required, a strong indication that the application is project-specific had been found as some level of customization will be required to use the notion in a project. The summary of the results is presented in Table 4.4.

To further strengthen future conclusions each domain expert was also asked if, based on their experience, the notion of trace link correctness could be applied to any project. The results are presented in Table 4.5. Further details on how it would be applied and what data would be needed to support the notion in any project was also collected.

- **Versioning**
  One domain expert stated that version numbers for each artifact and document exist in the analyzed dataset and that this data supports Versioning as a notion of trace link correctness. The dataset which supports Versioning was the dataset belonging to a project from the automotive company. The HIPAA project and MobSTr projects did not have any data supporting the notion of Versioning. All three domain experts agreed that information about Versioning in form of a version number on each artifact should be created in order to evaluate this notion of trace link correctness. One of the domain experts also identified that instead of creating Versioning on individual artifacts, it could also be possible to create a baseline of a project and apply Versioning on this baseline. This would help stakeholders identify the exact content of a specific baseline.

  "And then you either do that as individual artifacts or you would do that as some kind of a baseline and baseline meaning you know that you have a list, so in this baseline this is what's included. These are the artifacts that are in these versions are included in this baseline." (Interviewee 2)

  All participating domain experts agreed that Versioning is applicable on any project. The domain experts also agreed that the way to apply Versioning is by adding version numbers on all artifacts. One notable difference is that while two of the domain experts spoke of version numbers as pure numbers, the third domain expert explained that version numbers in the form of release dates would make more sense.

"Definitely the date when the last version was published or released. To make sure, I mean, if you notice that you are relying on an artifact that is very old, then you might could get suspicious why this artifact has never been updated recently, this could give give a hint that you might you might work with a with an old artifact, or a hint that no one is any more responsible for this artifact." (Interviewee 2)

The main motivation for this is that dates also add information about how well the artifact is being maintained. This is something not manageable with pure numbers for version handling.

All domain experts agreed that the basic data reuquired by all projects to apply Versioning was the same, i.e. version numbers applied to all artifacts. However, some potential customization need was found in the answers provided. Some projects may need version numbers on a baseline, i.e. a collection of a set of artifacts of a specific version number and the baseline itself would then have a version number of it's own. Furthermore, the data type of the version numbers might differ between the projects as simple numbers were suggested by two domain experts while dates as verion numbers was suggested by one domain expert. This is an indication that the base concept of the notion is generic regardless of the project, but some level of customization is needed making it a project-specific notion.

- **Lifetime/Lifespan**
  The same project which contained supporting data of Versioning was found to also contain supporting data to evaluate Lifetime/Lifespan. This was the project dataset from the automotive company. The notion is supported by a company-wide standard which contains specification on the Lifetime/Lifespan after the official publication or release date of all artifact types. Once again, all three domain experts agreed that the notion of trace link correctness could be applied to any project, but one of the domain experts expressed that this was not a notion of it's own but just another layer of requirements that should be handled.

  "The way I would set it up would be that it would be a requirement, as part of the requirement specification. There would be requirements related to lifetime and how long would the maintenance period be. These are important system- or project-level requirements to understand for the people developing the software or the project or whatever it is we are talking about. So I would definitely see that as just another layer of requirements." (Interviewee 2)

  All domain experts agreed that Lifetime/Lifespan could be used in any project but their opinion on how it should be applied differed. One domain expert argued that Lifetime/Lifespan should only be visible to the person who is re-

sponsible for maintaining the artifact or trace link. It is up to this responsible person to prolong the time period defined if needed. Users of the artifacts and trace links should only be bothered with the validity and not the dates themselves. As long as the person responsible for maintenance has prolonged the dates, the artifact and/or trace link should be shown as valid to the users.

"You don't have to explicitly mention the remaining lifespan of that link, you should maybe just have an indicator... I would rather have a person who owns that link and the owner of that link is responsible for maintaining it regularly and extending the lifespan if necessary. That has the advantage that there is a responsible person in the end. And this person of course, should get information about the remaining lifetime of the link. But the person who is a user of the trace link should only be informed about if the trace link is valid or not." (Interviewee 1)

Finally, the third domain expert argued that a company-wide standard should be available defining how long each artifact should remain stored after being published. This also implies that a publish date must exist and be stored for any released artifact.

"There should be a definition of all artifact types and their normal retention period within the project which is used to clarify how long the artifact should be maintained after the publish date. (Interviewee 3)

The domain experts agreed that some form of expiry date should be set to indicate when a trace link will no longer be maintained. However, how it is to be applied in a project requires some customization based on the project needs resulting in a project-specific classification of the notion.

- **Non-Duplicated Trace Links**
  With regards to analyzing the current datasets, none of the participating domain experts found supporting data in their current datasets. All three agreed that preventing duplication of trace links is difficult to achieve. While all agreed that some form of unique identifiers could be used, this would not be sufficient to avoid duplication. Two of the domain experts highlighted the same basic form of duplication bypassing the unique identifier protection mechanism, i.e., duplicating a trace link's contents into two trace links with unique identifiers. The same would be valid for artifacts.

  "I mean the links should maybe have a unique ID. But probably in a duplication, we'll get a new unique ID... I don't know how to mitigate this problem of avoiding duplication" (Interviewee 1)

  "Add unique ID on all trace links and artifacts. This could make it possible to identify trace links which connect the same artifacts. But again, even if

they connect different artifacts, the content of them could actually be duplicated thereby in a way making the trace link duplicated indirectly." (Interviewee 3)

An extension to the problem was also pointed out by the third participating domain expert. Additional detail to the problem was added by using requirements writing as an example. It is very difficult to write requirements that are not overlapping. This is not a case of duplication of trace data in the form of trace links, but rather a different form of duplication which could occur.

"I think that it's hard to make sure that you have the requirements that are not overlapping. Because I think the question sort of implies that you have perfectly written requirements and the implementation is very contained and so on. And usually that's not the case, there is some overlap" (Interviewee 2)

Non-Duplicated Trace Links was a notion which was difficult to classify. While none of the domain experts was certain that duplication can be fully avoided, the main suggestion which supports the notion in a project was to use unique identifiers on artifacts and thereby finding duplicated trace links. As this was applicable on all projects, the notion was classified as generic.

- **Unique Artifact Identification**
  With regards to analyzing the datasets, two domain experts stated that the dataset analyzed supports the notion of Unique Artifact Identification. The project from the automotive company had unique identifiers for every artifact class within the project and every identifier is also unique for each version with the artifact class prefix as part of the unique ID. The iTrust/HIPAA project was also identified as having unique identifiers for each artifact. However, an important issue was identified by the domain expert analyzing this dataset in that there is no support for different versions. This means that the unique identifiers would not be able to support multiple versions of each artifact. With the current data structure of the iTrust/HIPAA project, all versions of a specific artifact would share the same identifier.

  "At least as a static picture. If we look at when you start doing changes, and different versions, and so on, you need a unique identifier and the version, and then you will always be fine." (Interviewee 2)

  No supporting data was found in the MobSTr project dataset, but the domain expert suggested the addition of unique identifiers to all created artifacts through an automated mechanism.

  "Once you create an artifact, it should always come with an automatic ID with a unique ID." (Interviewee 1)

  All domain experts had suggested that unique identifiers should exist not only

identifying a specific artifact class, but also a specific version of the instanced artifact in the Data Model Level. However, some level of project-specific customization might be required. The project from the automotive company analyzed in this research, used prefixes based on artifact classes as a component of unique artifact IDs thereby making those identifiers different from other projects which will not have the same prefixes built into the artifact identifiers. As a result, Unique Artifact Identifiers was classified as project-specific.

- **Mandatory Artifact & Mandatory Trace Link**
  With regards to analyzing the current datasets, only one domain expert stated that data exists which supports Mandatory Artifact & Mandatory Trace Links as a notion of trace link correctness. This data was present in the form of identified legal requirements and standards that were being followed by the company. These additional requirements mandate the creation of specific artifacts and trace links in projects within the company.

  "We have a lot of legislation that mandates the creation of documentation and that this documentation is linked to different SW and HW part numbers in our end products. This is a form of artifacts and trace links that must be present in any projects sold on the specific markets. Also, the company follows a lot of standards, such as ISO26262, which requires other types of documents and trace links to be created and maintained during project development and during the maintenance phase". (Interviewee 3)

  With regards to applying the notion on any given project, some variation in opinion was found. An interesting quote from one of the domain experts highlights a basic summary of the difference between the participants and their opinions.

  "The consequences could perhaps be different, if you're breaking the law or are you just not having a satisfied customer, maybe that's the difference." (Interviewee 3)

  The quote summarises that the importance of having mandatory requirements and trace links may differ between projects. This depends on the consequence of specific artifacts and trace links not being created and maintained during project development or during the project maintenance phase.

  As different guidelines and regulations apply to different projects, this notion of trace link correctness was classified as project-specific.

Table 4.4: Summary of results for the question if the notions of trace link correctness exist in the analyzed project datasets followed by the question on how they are or could be applied

Notion	Interviewee 1 / MobSTr	Interviewee 2 / iTrust/HIPAA	Interviewee 3 / Automotive Company
Versioning	No, could add version number to artifacts	No, could add version number to artifacts or a baseline	Yes, version number on all artifacts
Lifetime/Lifespan	No, could add expiration date to artifacts and trace links and use these to show validity	No, add requirements that specify lifetime/lifespan	Yes, company-wide standard on all artifact types stating retention time after publication/release.
Non-Duplicated Trace Links	No, not sure what to add to avoid duplication	No, Versioning and ID should be added, but in the end human factor is involved	No, add unique IDs, but this does not prevent content duplication
Unique Artifact Identification	No	Yes, as long as each artifact exists in one version only. Multiple versions will not have unique identifiers	Yes, each artifact has a unique ID.
Mandatory Artifact & Mandatory Trace Links	No	No	Yes, there are a lot of identified legal requirements and standards in the company that are being followed. These have mandatory artifacts and trace links.

Table 4.5: Summary of results for the question if the notions of trace link correctness can be applied to any project and how they can be applied

Notion	Interviewee 1 / MobSTr	Interviewee 2 / iTrust/HIPAA	Interviewee 3 / Automotive Company
Versioning	Yes, add dates as version numbering	Yes, add version numbers to artifacts	Yes, add version numbers to artifacts
Lifetime/Lifespan	Yes, add expiration date to artifacts and trace links and use these to show validity	Yes, add lifetime on each version of trace link and artifact	Yes, add publish date to all artifacts and have a supporting document specifying retention time for all artifact types
Non-Duplicated Trace Links	Yes, add unique ID	Yes, add some form of unique identifier	Yes, add unique ID on all artifacts
Unique Artifact Identification	Yes, secure that all artifacts automatically get a unique ID	Yes, add unique identifiers and version numbers on all artifacts	Yes, add unique ID on all artifacts
Mandatory Artifact & Mandatory Trace Links	Yes	Yes and No. Depends or the system purpose	Yes, a guideline or specification summarizing all needed artifacts and trace links according to standards and legislation followed by the project

4.1.5 Generic or Project-Specific

To gain deeper understanding of how each notion should be classified, each domain expert was asked to further elaborate on their answers from the previous section. The focus was on understanding if the data suggested to support the notion could be applied in the same form in any project without the need for any project-specific customization. This data adds additional strength to the final conclusions when answering RQ2 and the results are presented in Table 4.6.

- **Versioning**
  With regards to Versioning, the opinion from all domain experts was that Versioning as a notion can be applied to any project. However, how it is applied and to which artifacts and trace links it applies to must be done in a customised way, i.e., project-specific. Two of the domain experts argued that there could be a generic process, model or description supporting the notion. Even if this is the case, what artifacts and trace links are to be covered depends on the project needs.

  "If you think about it in terms of how processes work... you can define and send information through generic information models, you can find a generic process for version control. For example, you can use the same process for version control, but then maybe in each case, it would be specific." (Interviewee 2)

  "You could use a generic definition, but what artifacts it applies to and how to apply Versioning must be project specific, because you might for instance use different systems for HW and SW development even within the same company." (Interviewee 3)

  As all domain experts had agreed that Versioning requires some customization before it can be applied and this agrees with the findings in the previous section, the final decision was to classify Versioning as project-specific.

- **Lifetime/Lifespan**
  With regards to Lifetime/Lifespan all three domain experts agreed that there could be a generic definition which can be used in any project. As was stated for Versioning, how Lifetime/Lifespan is applied must be customised based on the project needs, i.e., the application of the notion is project-specific. The following quote summarizes what all three domain experts stated:

  "The definition can be generic, but the lifespan itself might depend on the object that you have. My example was hardware and software. Hardware might have a design that might have a lifespan of 20 years. While software architecture might have a lifespan that is much shorter." (Interviewee 1)

Another interesting aspect to Lifetime/Lifespan was noted by one of the domain experts. There might be cases where there are active court orders preventing companies from deleting information while the court case is active. This might affect the expiry dates of relevant artifacts and trace links.

"Various types of artifacts will have different retention times. This will also depend on legislation and whether or not there are active court orders demanding extended retention times." (Interviewee 3)

As all domain experts had agreed that Lifetime/Lifespan requires some customization before it can be applied and this agrees with the findings in the previous section, the final decision was to classify Lifetime/Lifespan as project-specific.

- **Non-Duplicated Trace Links**
  For Non-Duplicated Trace Links all three domain experts agreed that this is a generic notion of trace link correctness and applicable on any project. This was further enhanced by all domain experts suggesting the same basic mechanism for supporting the notion, i.e., using unique artifact identifiers and in this way detect when multiple trace links exist with links to the exact same artifacts. Therefore the notion was classified as generic.

- **Unique Artifact Identification**
  Once more all domain experts agreed that this is a generic notion of trace link correctness that is applicable on any project. However, this is in conflict with the findings of the previous section. When collecting and analyzing the objective data in the previous section on how the notion should be supported by the different projects, it was found that the projects required different structures of the unique identifiers. Therefore the final decision was to classify the notion as project-specific.

- **Mandatory Artifact & Mandatory Trace Link**
  Mandatory Artifact & Mandatory Trace Links is another notion of trace link correctness where all domain experts agreed that definition of the notion is generic. However, two domain experts stated that the application must be project-specific.

  "The definition can be generic, but what is a compulsory tracelink and what is a compulsory artifact or mandatory artifact that is project dependent." (Interviewee 1)

  As a majority of the domain experts had agreed that Mandatory Artifact & Mandatory Trace Links requires some customization before it can be ap-

plied, the final decision was to classify the notion as project-specific.

Table 4.6: Summary of results on the question if the definition of the notion of trace link correctness can be generic or if it is project-specific

Notion	Interviewee 1	Interviewee 2	Interviewee 3	Final Result
Versioning	Project-Specific	Project-Specific, might be possible to have a generic version process in a company	Generic definition, but what artifacts it applies to and how must be project-specific	Project-Specific
Lifetime/Lifespan	Generic definition, but project-specific application	General definition, but specific application in a project	Generic description, but exact definition must be project-specific	Project-Specific
Non-Duplicated Trace Links	Generic	Generic	Generic	Generic
Unique Artifact Identification	Generic. Should be one ID creation system for the whole company	Generic	Generic	Project-Specific
Mandatory Artifact & Mandatory Trace Links	Definition is generic, but application must be project-specific	Generic	Generic in description, but must be project-specific in application	Project-Specific

4.2 Iteration 2

This iteration focused on answering RQ3: *Does evaluation of trace link correctness based on the identified notions require a domain-expert?*. The main goal was to understand if domain expertise was needed in order to evaluate trace link correctness using the identified notions. The secondary goal was to understand if experience level of engineers has an effect on the ability to evaluate trace link correctness using the identified notions. A questionnaire was sent to 15 participants and 12 responded. The respondents were 6 experienced and 6 non-experienced engineers within Software Engineering. The experienced engineers for this study were recruited from the industry. The non-experienced engineers were students within the Software Engineering field. For this iteration, only the iTrust/HIPAA dataset was used. The dataset from the automotive company was not included in this iteration due to secrecy of project data. The MobSTr dataset was omitted due to the limited time frame of this study. The answers from the test subjects were compared with the answers given by the domain expert in Iteration 1. Each answer was classified as being in agreement if it matched the answer by the domain expert. If the answer given differed, it was classified as being in disagreement. Furthermore, the experienced group and non-experienced group responses were treated separately in order to build further knowledge in the different levels of understanding between the two groups. Once the data had been analyzed, the artifact was updated. The results are summarized in this section and the final artifact is shown in Table 4.8 below.

4.2.1 Review of the artifact

The definition of each notion was written as a high level description. This was done to better explain the concept of each notion to the participants, thereby supporting the main purpose of Iteration 2, i.e., evaluating the level of understanding of the discovered notions of trace link correctness. The definition of Versioning was updated to contain information that teams can use the notion to detect that artifacts which are vital to their work have been updated. This enables synchronization in projects with multiple teams and ensures that teams are aware of changes done by other teams to important artifacts. The notion Lifetime/Lifespan got an extended definition including the possibility to set an expiration date to cover the entire lifetime of the product on the market. This was based on one of the domain experts stating that it must be possible to do this for some artifacts and trace links, particularly those that are mandated by legislation and various types of standards. The definition of Non-Duplicated Trace Links was updated to allow clones in a project. As was noted by the domain expert requesting this modification, it was clarified that an update to either the original or any of the clones should trigger the same update to all of them. Mandatory Artifacts & Mandatory Trace Links got the matching update that was given to Lifetime/Lifespan, i.e., that if both notions are used then the Lifetime/Lifespan should be set to cover the entire lifetime of the product on the market for all artifact and trace links that are mandatory. Unique Artifact Identification did not get an update in this phase as none of the domain experts had requested a change.
Only one update was made on the applicable level classification of the notions. Unique Artifact Identification was only classified as belonging to DML after Iteration 1 instead of both levels.
An additional column “Application” was added to the artifact containing information on whether the notion can be applied to any project in a generic way or if there must be some type of customization, thereby making it a project-specific notion. The updated artifact containing knowledge gained during Iteration 1 is show in Table 4.7 below.

Table 4.7: Updated List of Notions of Trace Link Correctness

Notion	Definition	Model Level	Application
Versioning	Versioning is about maintaining a change history log and securing that information about the evolution of each artifact is stored. As part of the concept, each change to any given artifact can be logged and this enables tracing changes via the version number to a certain point in time. Versioning allows teams working on multiple artifacts to show when an update has been released allowing other teams to react and thereby simplifies synchronization on multi-team project work.	DML	Project-Specific
Lifetime/Lifespan	Lifetime/Lifespan is about setting an end date to each trace link where maintenance will no longer be performed. It should be possible to set the end date to infinity for artifacts and trace links that must be stored for as long as the product developed remains in use in the market.	DML/IML	Project-Specific
Non-Duplicated Trace Links	Non-Duplicated Trace Links is about securing that there is no duplicated information in trace link data. This in order to avoid future update errors where only one link has been updated but not the duplicates. It is impossible to know which link is correct and which link was not updated correctly at some point in the project life cycle. If duplicated links are to be allowed then it must be possible to mark the duplicates as clones and when the either the original or one of the clones is updated, the change must be applied to all of them.	DML	Generic
Unique Artifact Identification	Unique ID is about being able to identify a specific artifact through it's ID. The ID only belongs to one single artifact in the project, thereby identifying it uniquely via the ID.	DML	Project-Specific
Mandatory Artifact & Mandatory Trace Links	Mandatory Artifacts & Trace Links is about securing that all artifacts and trace links that must exist according to used guidelines or regulation are created. Mandatory Trace Links is about securing that all trace links that must exist according to used guidelines or regulation are created. This secures that applicable guidelines and regulation are fulfilled by the project. Mandatory Artifacts and Trace Links must not expire and if Lifetime/Lifespan is used in the project they should remain for as long as the product is used on the market.	DML/IML	Project-Specific

4.2.2 Definition of the notions

In this section we present results from the questionnaire where the respondents were asked if they agree with the generic definition of each notion. The definition of each notion provided was the updated definition after Iteration 1, which was updated based on the feedback from the domain experts. These can be read in Table 4.7 above. If the respondents disagreed with the description, they were asked for what modifications they suggest.

- **Versioning**
  Results showed that most of the participants agreed on the definition of the notion, see Figure 4.1 below. However, one non-experienced respondent would modify the definition and rather use Versioning to extract information about changes. The example given was that it would be possible to see who did what change to an artifact or trace link and when the change was made. This is an extension to the concept of Versioning which was also discussed in literature by Gall et al. [42], where it was proposed that an automated versioning system would automatically increment version numbers when modifications were made to artifacts. The proposed versioning system would also add data on who and when introduced this change.

- **Lifetime/Lifespan**
  Disagreement regarding the definition of Lifetime/Lifespan was found amongst experienced respondents with regards to allowing infinite lifetime on artifacts and trace links. The opinion found on two respondents was that there should always be an end date which is controlled by standards and regulation after the product is no longer manufactured and/or used. Exact end time can vary depending on legislation for the product category. Interestingly, one of the participants from the experienced group agreed with the definition, but also proposed that data should be deleted once it has expired. This proposal is very similar to the third participating domain expert of Iteration 1, who stated that there should be a definition of how long artifacts are to be stored after publication within the company. The results are presented in Figure 4.2 below.

- **Non-Duplicated Trace Links**
  Figure 4.3 below, shows that no deviation of opinions was found amongst any of the groups with regards to the definition of the notion of Non-Duplicated Trace Links.

- **Unique Artifact Identification**
  Results showed that most of the participants agreed on the definition of the notion, see Figure 4.4 below. The only response deviating was from the experienced group. The respondent highlighted that the description should also include that unique artifact identifiers should exist on the group level. The suggestion was focused on adding unique identifiers on a group of artifacts, such as a baseline. This baseline could for example be a software release con-

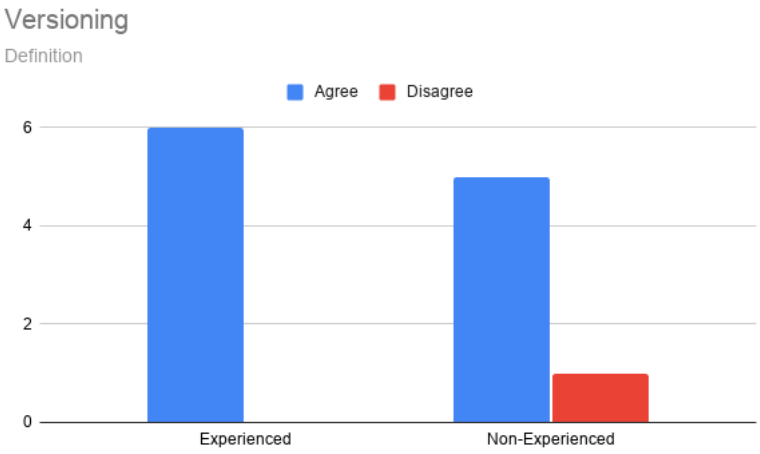

Figure 4.1: Results on the question if the participants agree with general definition of Versioning

taining a number of artifacts of a specific version. A similar opinion was found when the domain expert was asked on what data needs to be added to support Versioning as a notion.

- **Mandatory Artifact & Mandatory Trace Links**
  One of the experienced respondents disagreed with the definition of the notion and highlighted that artifacts and trace links can expire even if the product is still in use on the market. There can be legislative requirements defining how long artifacts and trace links must be stored and this time could be shorter

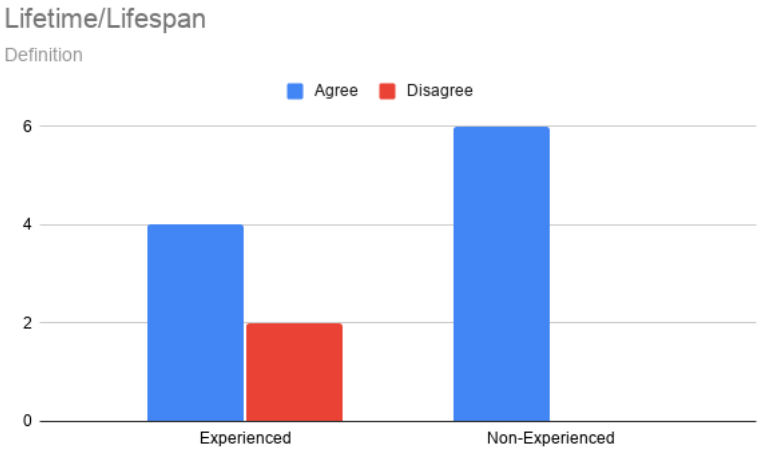

Figure 4.2: Results on the question if the participants agree with the general definition of Lifetime/Lifespan

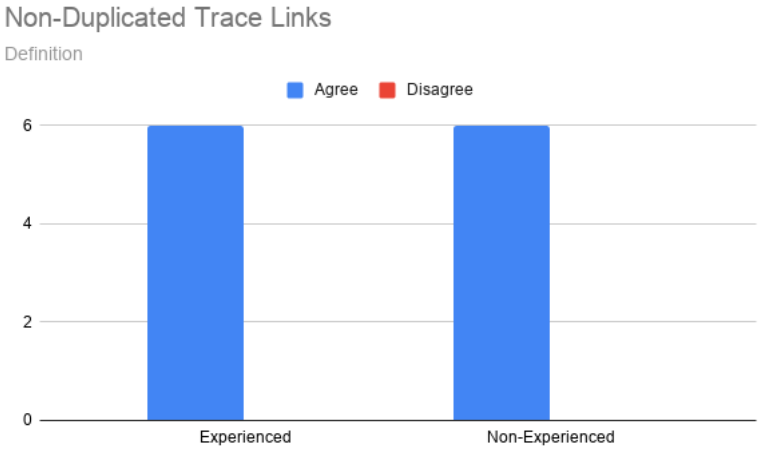

Figure 4.3: Results on the question if the participants agree with the general definition of Non-Duplicated Trace Links

than the lifetime of the product on the market. The results are presented in Figure 4.5 below.

4.2.3 Existence of the supporting data in the project

In this section we present results from the questionnaire where the respondents were asked if they can evaluate the notion of trace link correctness with regards to the analyzed project dataset from iTrust/HIPAA. If supporting data was found in the dataset, the respondent got a follow-up question asking what data they had identi-

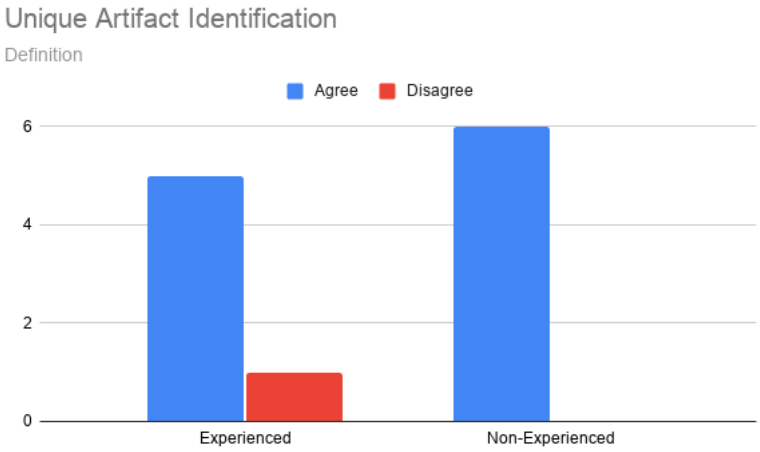

Figure 4.4: Results on the question if the participants agree with the general definition of Unique Artifact Identification

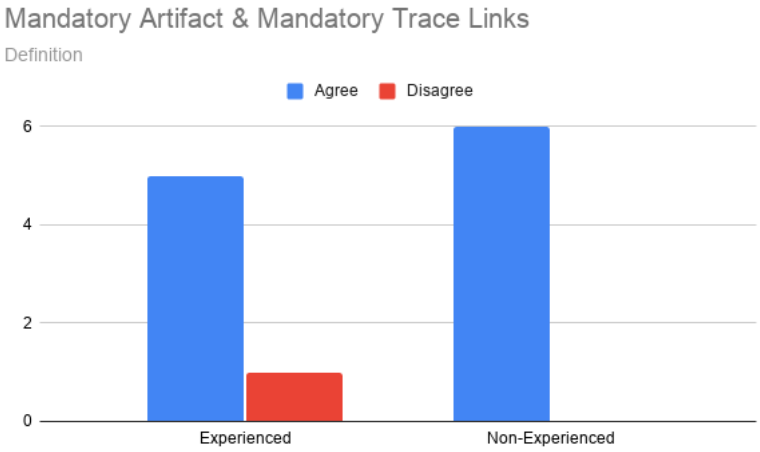

Figure 4.5: Results on the question if participants agree with the general definition of Unique Artifact Identification

fied. The responses were compared to the response of the domain expert provided when analyzing the same dataset during Iteration 1. If the respondent had provided the same answer as the domain expert, the answer was classified as being in agreement. However, if the answer did not match the answer provided by the domain expert, it was classified as being in disagreement.

- **Versioning**
  In Iteration 1, the domain expert had stated that the dataset did not contain data that supports evaluation of trace links according to Versioning. One clear misconception was noted with regards to finding data supporting Versioning amongst the respondents. Numbers used in the dataset were interpreted as version numbers instead of an identification number. For example, the requirement iT6.1 is a sub-requirement to requirement iT6. However, this was misinterpreted as iT6.1 being a later revision of requirement iT6. The misinterpretation was more common amongst the non-experienced respondents. These results were treated as not being in agreement as they did not match the answer given by the domain expert in Iteration 1. All other participants had provided an answer that agreed with the domain expert. The results are presented in Figure 4.6 below.

- **Lifetime/Lifespan**
  With regards to Lifetime/Lifespan, the domain expert had stated that the dataset contained no supporting data for evaluation of trace link correctness. One respondent had found data supporting Lifetime/Lifespan in the analyzed dataset. Upon further analysing the response it seems that the respondent had analyzed the project requirement set. Upon close inspection, requirements in

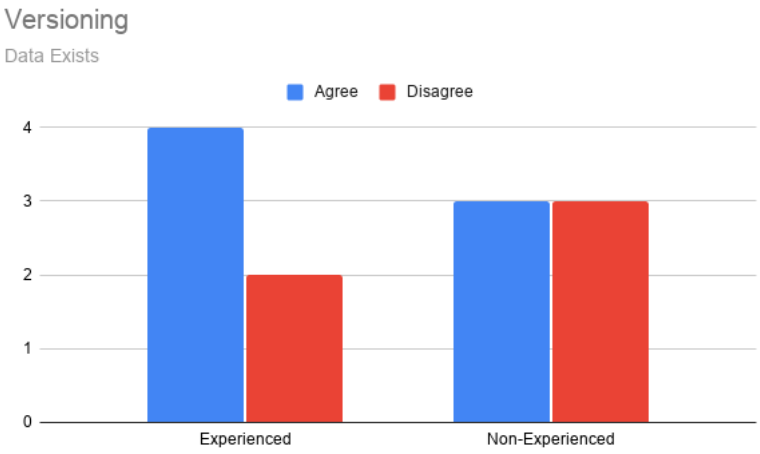

Figure 4.6: Results on the question if the participants can evaluate trace link correctness with regards to Versioning

the analyzed project data requirement set define Lifetime/Lifespan on logs produced by the system which should stop once user is not authenticated. This was the only result classified as not being in agreement, and the total results are presented in Figure 4.7 below.

- **Non-Duplicated Trace Links**
  Four of the experienced respondents agreed with the domain expert that no data supporting this notion was found in the analyzed dataset, meaning that it is not possible to identify duplication. However, two of the respondents claimed to have found supporting data. Upon close inspection of the answers, it was found that the supporting data was not in the trace data but in the artifacts of the project. In this particular case, there were requirements in the project where duplication of some specific data was not to be allowed by the developed system. As these requirements were not set on the trace data, but on the developed system, the answers were classified as not being in agreement with the domain expert. The non-experienced group had a respondent finding supporting data via the naming convention of the artifacts, which is similar to the proposed extension of the dataset by the domain expert, i.e., adding unique identifiers to each artifact. The answer was classified as not being in agreement with the domain expert. Another respondent amongst the non-experienced group misread the data and used requirements as something that is not an artifact. In this case it was suggested that the requirements on each artifact make the artifact unique compared to other artifacts. These answers was classified as not being in agreement with the domain expert. The results are presented in Figure 4.8 below.

- **Unique Artifact Identification**
  While the domain expert had found data supporting Unique Artifact Iden-

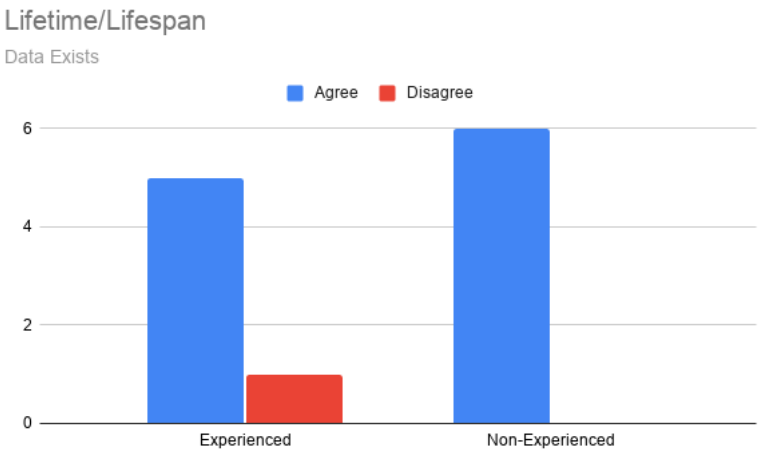

Figure 4.7: Results on the question if the participants can evaluate trace link correctness with regards to Lifetime/Lifespan

tification in the dataset of the project, one respondent from each group had not. Additionally, one of the respondents in the experienced group looked in the project data requirement set instead of the trace data. In this case, the set included a requirement that each system user or entity should have a unique user identification. This misinterpretation was the same as noted on the responses given on the notion of Non-Duplicated Trace Links. These answers were classified as not being in agreement with the domain expert. The remaining respondents had not only stated that there was supporting data in the dataset, but had also identified the same data as the domain expert. As they had agreed with the domain expert that supporting data exists and also pointed at the same supporting data, the answers were classified as being in agreement. The results are presented in Figure 4.9 below.

- **Mandatory Artifact & Mandatory Trace Links**
  Unlike the domain expert, one of the experienced respondents found supporting data in the dataset for Mandatory Artifact & Mandatory Trace Links. When reviewing the response in detail, it was clear that the project requirement set and not trace data had been used to identify this supporting data. The response had references to requirements and no reference to trace data was provided. The same type of misinterpretation was found in two of the responses from the non-experienced group. The results are presented in Figure 4.10 below.

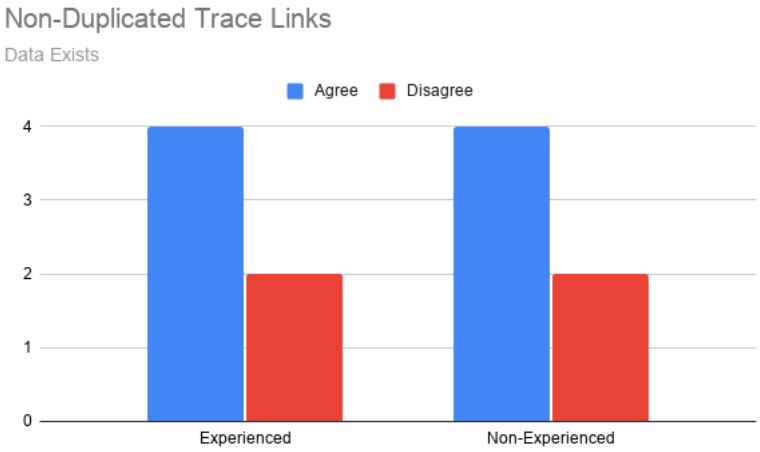

Figure 4.8: Results on the question if the participants can evaluate trace link correctness with regards to Non-Duplicated Trace Links

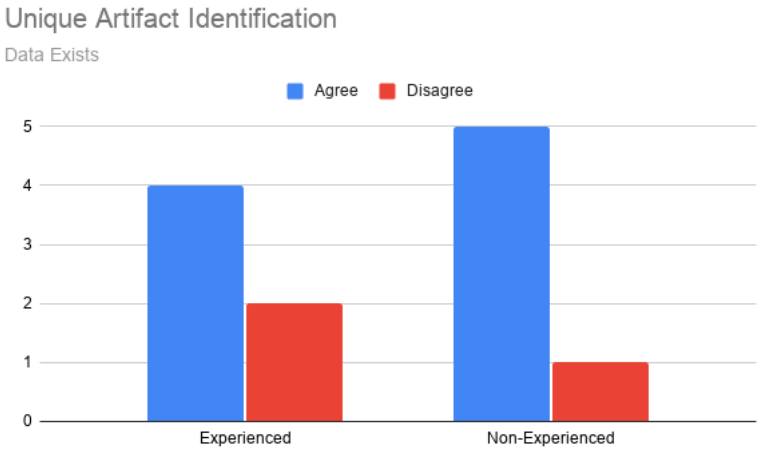

Figure 4.9: Results on the question if the participants can evaluate trace link correctness with regards to Unique Artifact Identification

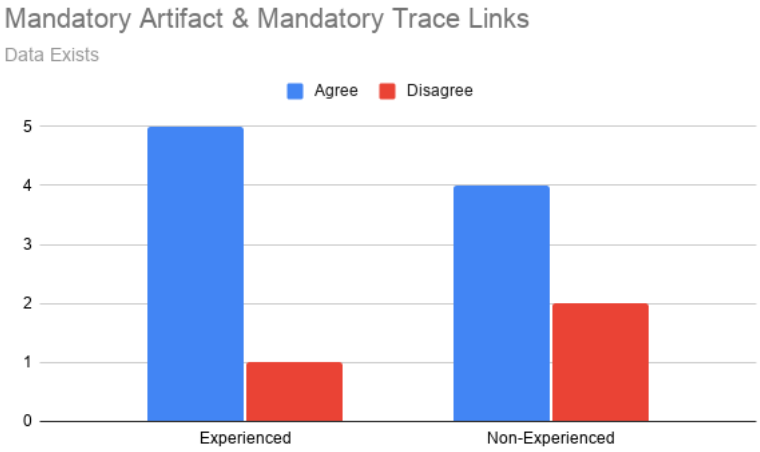

Figure 4.10: Results on the question if the participants can evaluate trace link correctness with regards to Mandatory artifact & Mandatory Trace Link

4.2.4 Extending the dataset to support the notion

In this section we focus on the responses that did not find supporting data in the previous question, as presented in Section 4.2.3 above. As some of the respondents had found supporting data previously, only the remaining responses were analyzed as part of this section. If the respondent had proposed the same data extension to support the notion as the domain expert did during Iteration 1, the answer was classified as being in agreement. However, if different data was proposed, the answer was classified as being in disagreement. This data provides insights whether the respondents understand on how to apply the notion or not.

- **Versioning**
  The domain expert had proposed that the data would have to be extended with version information to artifacts during Iteration 1. Only one deviating response was found during Iteration 2 and this was in the group of non-experienced engineers. The proposed mechanism would be to highlight what was changed between release versions of the product in the Information Model. This suggestion seems to assume the data model can only contain one instance of each artifact present in the information model. This answer differed considerably from the one given by the domain expert during Iteration 1 and was therefore classified as being in disagreement. The results are presented in Figure 4.11 below.

- **Lifetime/Lifespan**
  Four out of five experienced respondents that had not found supporting data in the dataset, proposed adding data in the form of a time factor, end date or similar, which matched the response from the domain expert. One experienced

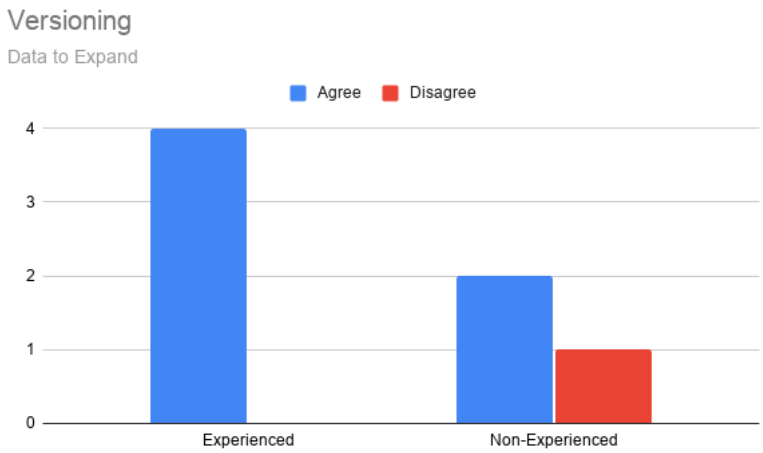

Figure 4.11: Results on the question of what data needs to be added in order to support Versioning

respondent was not sure what to add. Amongst the non-experienced respondents, one out of six had provided an answer matching the domain expert. One of the respondents of the non-experienced group suggested using coloring on the information model as a way to signify artifacts with no end date, but had no proposal on how to set an end date on artifacts or trace links which actually have an end date. This proposal was found too deviating from the proposal provided by the domain expert during Iteration 1 and was therefore classified as not in agreement. The results are presented in Figure 4.12 below.

- **Non-Duplicated Trace Links**
  All respondents from the experienced group that had suggested that data needs to be added to the dataset, stated that the dataset can be extended in order to do so. They proposed the same data to add as the domain expert, which was to add an attribute that can be used to either prevent the creation of duplicates or identify the existence of them. The main difference was that one respondent also added that specific trace links should be added between original and cloned artifacts. All of these answers were classified as being in agreement, as the one additional proposal was found to be within the scope of the proposal by the domain expert. In the non-experienced group the answers provided focused on using automated tools and coloring to highlight non-duplicates. These answers were classified as not being in agreement as they did not provide information on what data should be added to support the notion for trace link evaluation with regards to the notion. The results are presented in Figure 4.13 below.

- **Unique Artifact Identification**
  Both the experienced and non-experienced respondents that had not found

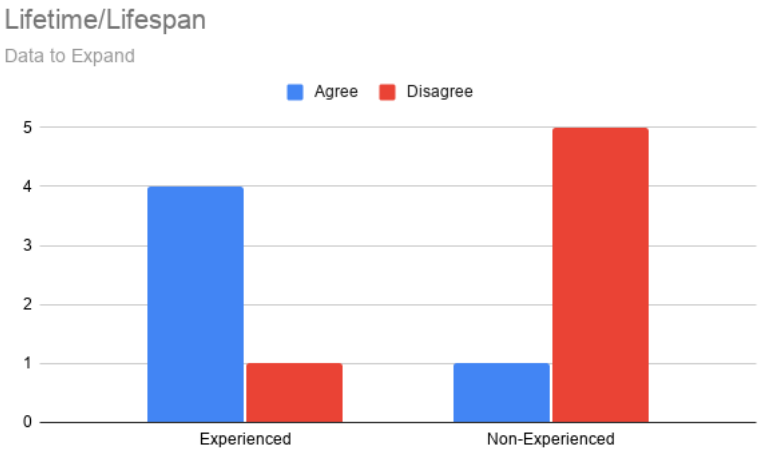

Figure 4.12: Results on the question of what data needs to be added in order to support Lifetime/Lifespan

data supporting Unique Artifact Identification in the dataset, proposed to add sequential numbers in the identifier of each artifact. The proposal matches the data identified as existing and supporting Unique Artifact Identification by the domain expert stated during Iteration 1. It is unclear why these two respondents failed to identify the data in the analyzed dataset. The results are presented in Figure 4.14 below.

- **Mandatory Artifact & Mandatory Trace Links**
  One of the experienced respondents had a different opinion of what data to add

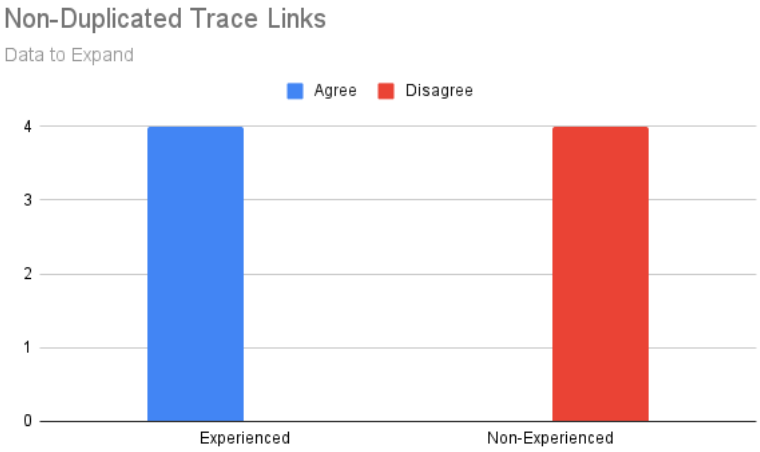

Figure 4.13: Results on the question of what data needs to be added in order to support Non-Duplicated Trace Links

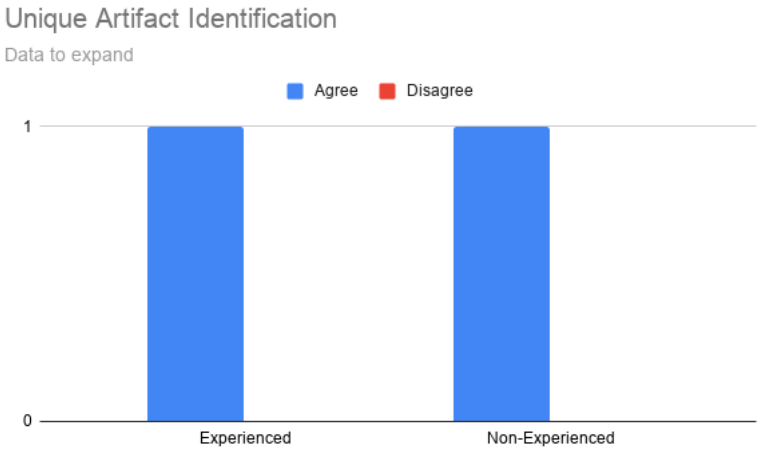

Figure 4.14: Results on the question of what data needs to be added in order to support Unique Artifact Identification

when comparing to the domain expert and the rest of the group. The deviation found was that the respondent was of the opinion that the TIM should contain a definition of what artifacts and trace links are mandatory. This is not a direct disagreement with what was stated by the domain experts, as they had pointed out that the notion was applicable to both levels, but no details were provided by the respondent on what data to add in the Data Model Level. For this reason, the response was classified as not being in agreement with the domain expert. Within the group of non-experienced respondents, three respondents highlighted that there should be an extension on the Information Model Level showing what artifacts and trace links are mandatory. Again, this is not in direct disagreement but they could not identify what data to add to the Data Model Level. One respondent of this group assumed that knowledge of what is mandatory was well known and that using audits would secure that the mandatory artifacts would be created. All responses of this group were classified as not being in agreement with the domain expert. The results are presented in Figure 4.15 below.

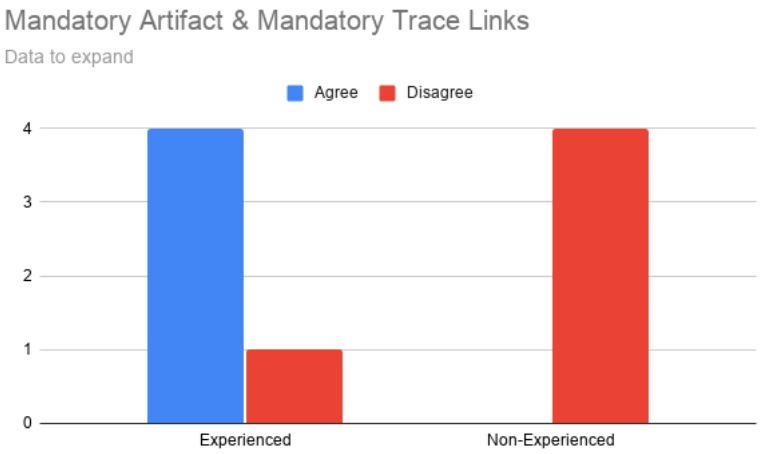

Figure 4.15: Results on the question of what data needs to be added in order to support Mandatory artifact & Mandatory Trace Link

4.3 Final artifact

The definition of each notion of trace link correctness was updated to clearly state how the notions makes a trace link correct.
The data collected in Iteration 2 showed that there is a low level of understanding on how to use the identified notions of trace link correctness amongst the non-experienced participants of the study, with the sole exception of the notion Unique Artifact Identification. The experienced participants had shown a good level of understanding for all notions and there was a good level of correlation between the results from the domain experts and the experienced participants of the study. This indicates that domain experts are not required to evaluate trace link correctness using the identified notions for these specific projects. To reflect this gained understanding, the column "Minimum Experience Level" was added to the artifact. This column contains information on the minimum experience level required to evaluate the correctness of trace links using data supporting the identified notion. The possible experience levels defined in this study are Domain Expert, Experienced Software Engineer or Non-Experienced Software Engineer. Unique Artifact Identification was classified as not needing any level of expertise by developers in a project, while the other notions were classified as requiring Experienced Software Engineers. This should help future trace strategy planners to identify which notions can be useful in a project depending on the involved development team compositions.

Table 4.8: Final artifact

Notion	Definition	Evaluation of trace link correctness	Model Level	Minimum Experience Level
Versioning	A TDM is correct if each artifact and trace link receives a new version number after each change. Additionally, each change, who performed it and when is stored in a history log. Data linking each artifact and trace link to it's corresponding history log must exist.	A trace link is correct if it links to the latest version of each linked artifact. If a later version of a linked artifact exists, the trace link must be reviewed for decision if it is to be updated or removed.	Project-Specific	Experienced Software Engineer
Lifetime/Lifespan	A TDM is correct if each trace link has an end date set defining when maintenance will no longer be performed.	A trace link is correct if it's maintenance end date has not passed.	Project-Specific	Experienced Software Engineer
Non-Duplicated Trace Links	A TDM is correct if there is no duplicated information in trace link data. If duplicated links are allowed, then it must be possible to mark the duplicates as clones and when the either the original or one of the clones is updated, the change must be applied to all of them.	A trace link is correct if there are no duplicates of the trace link or if all duplicates are marked as clones.	Generic	Experienced Software Engineer
Unique Artifact Identification	A TDM is correct if it is possible to identify a specific artifact through it's ID.	A trace link is correct if all linked artifacts can be uniquely identified using their ID.	Project-Specific	Non-Experienced Software Engineer
Mandatory Artifact & Mandatory Trace Links	A TDM is correct if all created artifacts and trace links which are mandated by applied guidelines or regulations, are created and traced in accordance to those guidelines or regulations.	A trace link is correct if it links to required artifact types in accordance to the applied guidelines or regulations.	Project-Specific	Experienced Software Engineer

5
Discussion

In this section, we discuss the results with focus on the research questions. Section 5.1 discusses results related to RQ1 and the identified notions of trace link correctness. How these notions can be applied to projects and how generic they are is addressed in Section 5.2. Finally, Section 5.3 addresses considerations on required domain expertise when evaluating trace link correctness using the identified notions.

5.1 Notions of Trace Link Correctness

This work focused on finding and understanding notions of trace link correctness as the community does not have the same view on what defines trace links as correct. In the literature one of the most common mechanisms to test the correctness of a trace link is to consult a domain expert [44]. However, this has been found by other studies to be of low reliability as human feedback cannot be fully trusted [5]. In an attempt to assist practitioners in the field and future research work we set *RQ1: What notions define trace link correctness?*. To answer this, it was important to distinguish trace link types on the information model and the trace links on the data model. Rempel et al. and Mäder et al. define the Traceability Information Model as a model which defines types of artifacts to be traced and their relations [12, 13], while the Traceability Data Model is defined as a representation of all traceability data created and maintained throughout the product life cycle [13]. Findings of this study confirmed that considering these two levels is important as some notions of trace link correctness clearly belonged to one of the two levels, implying that some of them are useful when constructing the project TIM, while others are useful when creating and maintaining trace data of a project. The results of this research distinguished notions which define correctness of trace links on these two model levels and focused on the TDM level.

Versioning, Non-Duplicated Trace Links and Unique Artifact Identification were found to define correctness of trace links on the Data Model Level. Interestingly, Lifetime/Lifespan and Mandatory Artifacts & Mandatory Trace Links were found to be applicable both on the Data Model Level and the Information Model Level. The identified notions Link Type, Granularity, Purpose, Non-Redundancy of Traceability Paths, Arity and Directionality were found to be applicable only on the Information Model Level. As this research focused on the Data Model Level, these notions are not further discussed. However, it should be noted that when constructing the project TIM, all notions that were found to be applicable on the Information Model

level should be considered as they can potentially provide value.
The only contradiction found in our study was that literature had suggested that Unique Artifact Identification should also apply to the Information Model Level. The proposal found in literature was not directly contradicted by any findings of our study and it is possible that none of the participants considered the solution proposed by Mäder et al. [14].

5.2 Generic or Project-Specific Notions

This section focuses on addressing results relating to *RQ2: Are the defined notions of trace link correctness generic or project-specific?*. The research question focused on understanding if the found notions of trace link correctness are determined to be project-specific, where the application of the notion must be customized based on the project needs, or if the notions can be generalized, and are thereby applicable to every project in a generic way. If generalization was found possible, tools for automatic evaluation of trace link correctness could be developed without further need of customization to fit specific project needs thereby decreasing the need of human involvement [8, 9, 10, 11].

Versioning, Lifetime/Lifespan, Unique Artifact Identification and Mandatory Artifacts & Mandatory Trace Links were found to be generic in their definition but their application must be customized according to the project-needs when applied. This makes all four notions project-specific as the results of this study indicates that there is no generic application technique that can be applied on any project for these notions.
Non-Duplicated Trace Links is a notion of trace link correctness that was found to be generic. This applies to all aspects of the notion such as what model level it applies to, how it is defined and how it is applied. No evidence was discovered indicating that there is a project-specific aspect to this notion, neither in the literature or in the results of this study. This means that it could be applied to any project using the definitions and application techniques proposed. It should be noted that no evidence was found during this study which would prove that the proposed application will protect against duplication of trace links in a project.

The major issue found with applying Versioning was a high risk of tediousness, as it was highlighted by one of the domain experts. It was noted that using Versioning on all artifacts and trace links in a project might get complicated with regards to maintenance. However, a solution to this is proposed in the literature by Gall et al., as this tediousness could be countered by using an automated Versioning Control System (VCS) [42]. Such a system would remove the maintenance effort, thereby removing the major identified obstacle found in this study to using Versioning on the Data Model Level. It is possible that the domain expert that expressed this concern is active in a domain where using VCS is not a common practice.
An interesting aspect of Versioning was provided by Gall et al. who suggested that additional data should be stored in the version information of an artifact, where at least author and change dates should be included [42]. In our research, several

practitioners supported this approach as it would allow use of trace data to identify abandoned trace links that have not been maintained for a long time and are at a high risk of being invalid.
Another interesting finding of this study was that Versioning should be applied to baselines and not only artifacts and trace links in order to support users of trace data to identify which version of a specific artifact was included in a major product release. This can be regarded as confirming suggestions by Anquetil et al. where the mechanism could be used to trace backwards from a specific baseline release, such as a beta release, to understand what requirements and features were included in that specific product release [47]. The need of baselines was not identified by all participants, which indicates that not all domains use baselines in their standard development process.
As using a VCS to support the application of the notion is possible, it is a strong indication that Versioning, as a concept or a process, can be generic. However, what artifacts it applies to and how it is used in a project is project-specific and therefore which VCS is used depends on the needs of the project.

Lifetime/Lifespan was a notion of trace link correctness which was mentioned in literature in various forms [3, 43], but an actual mechanism to implement it was not found. In our study we identified multiple approaches to support this notion of trace link correctness. This is a strong indication that Lifetime/Lifespan is a project-specific notion. It is possible that the notion has been expressed as a concept in theoretical studies in literature, but never studied in practical use cases. Our study found practical cases where the notion was applied and could therefore extract the mechanisms used.
Cleland-Huang et al. [43] focused on reducing maintenance effort once the need of certain trace links diminished, and it was the main reason provided of why the notion should be used. However, as discovered in this study, being able to define that some artifacts and trace links must be maintained for some time even after project completion is a different purpose of the notion. This could be a reason why multiple mechanisms were proposed and why we failed to find a generic application technique which could be acceptable on any project.
An interesting finding in the study was that the company which applied the notion had a standard defining retention and maintenance time after publication date. The standard only defined times for artifacts and not trace links. This implied that while artifacts are treated as important data, trace links might not be, which indicated that the company was not treating traceability as an important tool for success. As the company followed the ISO26262 standard, which does mandate some mandatory trace links that should be retained and maintained even after the product launch, it could indicate that the company treated mandatory trace links in a fashion similar to the proposal of another participating domain expert, i.e., as a separate layer of requirements. This would imply that the mandatory trace links were setup as requirements and thereby treated as artifacts instead and in this way getting a defined retention and maintenance period through the company standard.
One major implication found in our study was the possibility of court cases preventing companies from deleting data. This includes both artifacts and trace links.

If Lifetime/Lifespan is used as a notion of trace link correctness in a project, the possibility of court cases overriding possible expiry dates should be considered.
Another major implication for practitioners was that whatever mechanism is used in the background, users of trace data should only see if the trace link is valid or not. The person responsible for maintenance of a trace link should be the only person actually seeing a potential expiry date. As long as the trace link is not expired, users should see it as valid and if it has expired they should see it as invalid. This finding has a major impact on how a traceability tool should be implemented.

Another finding in this study was that a duplication in the form of clones might be needed in some projects. This is a contradiction to the proposal by Mäder et al, who proposed that no duplication should be allowed [14]. However, it should be noted, that it was also identified in this study that modifying the original or any of the clones should affect both the original itself and all clones. This means that the cloning process could be implemented just as Mäder et al. suggested, i.e., storing all artifacts and trace links in a database-like repository and removing duplicates on regular basis. The clones would instead be supported by the visual layer only showing the clones to the users, but in the storage layer, the database would only keep one single copy of each artifact and trace link. We assume that the purpose of this type of structure could be to provide visual filters where the original artifact or trace link shows one set of data and the clones show other types of data, such as comments with a filter setting activated. In the database only the super-set entity is stored containing all data.
One major problem in using this notion of trace link correctness in a project was found in this study. While it is possible to find duplicates of trace links in a database-like repository using unique identifiers as the search criteria, it was noted that this is insufficient protection against duplication. In the end, human users could create new artifacts with unique IDs but the same content.

Mandatory Artifacts & Mandatory Trace Links notion of trace link correctness was also found to have a generic definition, but an application which must be customized based on the project needs. This makes it a project-specific notion of trace link correctness. Rempel et al. identified how the notion would be applied was highly dependent on what regulations and standards apply to the project [21]. This was reinforced by the findings in this study.

5.3 Usage of Notions of Trace Link Correctness by Non-Domain Experts

The goal of this work was to also understand if correctness of a trace link must be determined by consulting a domain expert as stated in [3, 1]. This was the main purpose of *RQ3: Does evaluation of trace link correctness based on the identified notions require a domain expert?*.

Unique Artifact Identification was the notion of trace link correctness which was found to have the greatest level of understanding regardless of experience level. No domain expert is needed and no prior professional experience for the user to understand how to evaluate this notion.

With regards to the other notions of trace link correctness an unexpected discovery was made. None of them really need domain experts for evaluation. What is needed though is a certain level of professional experience. Disregarding some minor misunderstandings on what data parts to analyze during the study, the participating professionals had a high level of understanding of each notion and successfully evaluated all notions. This cannot be said for the participants with a low level of experience. What came as a major surprise was that the notion of Versioning was not well understood at all by this group of test subjects. The researchers had expected that with the environment today, where software and applications are regularly updated with new version numbers after each update, the non-experienced group of test subjects would be very familiar with the concept of Versioning. The expectation was that they would be able to evaluate the notion to a level similar to that of a domain expert. Furthermore, considering that team based software development is a large part of the education where multiple users push their latest updates to a central master, it should not be a new concept to handle various versions.

Lifetime/Lifespan, Non-Duplicated Trace Links and Mandatory artifact & Mandatory trace links were not understood at all. Even if the group had received a description of each notion beforehand, they were unable to grasp how these notions could be evaluated on the provided dataset. This was a major difference when compared to the group of experienced test subjects. It is clear that these are concepts that students do not get familiar with during their education period, but seasoned professionals recognise the problems and understand what mechanism can be used to prevent them.

In the case of Lifetime/Lifespan, a likely reason for the lack of understanding could be that school projects tend to be completely terminated once completed. Usually, there is no maintenance phase where parts of the documentation, trace links or other project data must be maintained. With regards to Non-Duplicated Trace Links, it could be the case of not containing sufficient levels of traceability requirements on school projects. This study showed that students had no understanding of the concept at all, indicating that they have not run into the problem of duplicated trace links, and possibly traceability in general, during their education period.

Mandatory Artifact & Mandatory Trace Links was also a concept which was found to be poorly understood by students. It comes as somewhat of a surprise, as mandatory artifacts are quite common during the education period. It is not uncommon to have both code, documentation and other artifacts as mandatory deliveries for a school project. This indicates that students are not well aware of what artifacts and trace links are, indicating a weak understanding of the core concept of traceability itself.

While the work done by Salman et al. showed that students are representative of professionals in software engineering experiments [48], our study results indicate the opposite. It is possible that while students can perform similarly to professionals in terms of code quality metrics, as proven by Salman et al., our study demonstrates that their level of understanding of traceability is well below that of professionals. This indicates that software engineers learn about traceability through practical experience in their professional career and not through their studies. The main implication of this is that students should not be used to represent professionals in studies involving traceability.

6
Conclusion

In this research, we identify what notions of trace link correctness exist in order to evaluate trace link correctness. This aids practitioners in the field of traceability when creating and maintaning trace datasets and performing future studies in the field. We also investigate if the found notions are generic or if their application must be customized for every project, making them project-specific. To conduct the investigation, an exhausting search in the literature for potential notions of trace link correctness was performed. The gained knowledge was evaluated in two iterations. The first iteration focused on understanding which notions can define a correct trace link and if the application of these notions must be customized when used in a project. The focus of the second iteration was to understand if evaluation of trace link correctness using the identified notions requires a domain-expert.

Of the 11 candidate notions of trace link correctness found in literature, the study showed that five notions are applicable on to the Data Model Level. These are the notions of Versioning, Lifetime/Lifespan, Non-Duplicated Trace Links, Unique Artifact Identification and Mandatory Artifact & Mandatory Trace Links.

One notion of trace link correctness is generic. The notion of Non-Duplicated Trace Links was identified as a generic notion of trace link correctness.
The remaining notions of trace link correctness require a degree of customization based on the project needs before they can be applied and are therefore classified as project-specific. Mandatory Artifact & Mandatory Trace Links typically applies to artifacts and trace links which are required by legislation and standards that apply on a project. This means that the application must be customized accordingly.

Of the notions found, only Unique Artifact Identification was found to be properly understood by most participants of Iteration 2. The conclusion is that no form of expertise is needed in order to understand and evaluate trace link correctness using the notion. While domain expert is not required to evaluate trace link correctness based on Versioning, Lifetime/Lifespan, Non-Duplicated Trace Links and Mandatory Artifact & Mandatory Trace Links, a higher level of experience is required. The non-experienced participants of this study failed to understand the concept of these notions and were unable to understand how they could be applied to the analyzed project. With regards to Versioning, it requires a domain expert to some degree. While the notion is well understood as a concept and how it should be applied, a lot of confusion was found amongst the participants when project trace data was analyzed. The identification numbers used in the analyzed project, caused

misinterpretation primarily amongst the non-experienced participants. Some level of misinterpretation was also found amongst the experienced engineers.

A conclusion is that experience is an important factor in understanding the found notions of trace link correctness. Domain expertise is not required, but rather experience within software development in general. Interestingly, the study indicates that students within the Software Engineering field are ill prepared for the reality that awaits in practice. The level of understanding of basic concepts, such as duplication and versioning seem to be poorly understood. These are problems professionals deal with during their careers, as can be seen by an improved level of understanding with a higher level of experience, but students seem to lack knowledge in this field when they graduate. This should serve as a wake-up call to Universities that insufficient focus is put on traceability during the education period and researchers should be wary when they recruit for studies within traceability.

6.1 Future Work

The research study only involved test subjects with a software engineering background. As traceability is a field involving multiple disciplines in the industry, a further study to understand how the identified notions of trace link correctness are understood and can be applied by practitioners from other fields would extend the knowledge gained during this study.
Given the limitations of the study, a future study could focus on how generalizable the results from this study are by including multiple projects from multiple domains. In a larger scale study, it would also be advisable to include multiple experts from each project to avoid suject bias.

The identified notions of trace link correctness could be further evaluated in their effectiveness in supporting practitioners when maintaining trace links. This could be a future study which could provide additional knowledge on how useful each identified notion of this study is in practical use.

Several notions of trace link correctness that were identified in this study, were found to be applicable on the Information Model Level. A future study should focus on how these notions can be used when constructing the project traceability information model and measure the usefulness of each. This would expand the knowledge base of how to construct effective TIMs in projects and aid practitioners when creating and maintaining project trace data.

Bibliography

[1] Andrea De Lucia, Fausto Fasano, Rocco Oliveto, and Genoveffa Tortora. Recovering traceability links in software artifact management systems using information retrieval methods. *ACM Transactions on Software Engineering and Methodology*, 16(4):13, sep 2007.

[2] K. Jaber, B. Sharif, and C. Liu. A study on the effect of traceability links in software maintenance. *IEEE Access*, 1:726–741, 2013.

[3] S. Maro, A. Anjorin, R. Wohlrab, and J. Steghöfer. Traceability maintenance: Factors and guidelines. In *2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 414–425, 2016.

[4] Orlena Gotel, Jane Cleland-Huang, Jane Huffman Hayes, Andrea Zisman, Alexander Egyed, Paul Grünbacher, Alex Dekhtyar, Giuliano Antoniol, Jonathan Maletic, and Patrick Mäder. Traceability fundamentals. In *Software and Systems Traceability*, pages 3–22. Springer London, oct 2011.

[5] Jane Cleland-Huang, Orlena C. Z. Gotel, Jane Huffman Hayes, Patrick Mäder, and Andrea Zisman. Software traceability: Trends and future directions. In *Future of Software Engineering Proceedings*, FOSE 2014, page 55–69, New York, NY, USA, 2014. Association for Computing Machinery.

[6] A. Egyed, K. Zeman, P. Hehenberger, and A. Demuth. Maintaining consistency across engineering artifacts. *Computer*, 51(2):28–35, 2018.

[7] Ralf Dömges and Klaus Pohl. Adapting traceability environments to project-specific needs. *Commun. ACM*, 41(12):54–62, December 1998.

[8] Jin Guo, Natawut Monaikul, and Jane Cleland-Huang. Trace links explained: An automated approach for generating rationales. In *2015 IEEE 23rd International Requirements Engineering Conference (RE)*. IEEE, aug 2015.

[9] Nan Niu and Anas Mahmoud. Enhancing candidate link generation for requirements tracing: The cluster hypothesis revisited. In *2012 20th IEEE International Requirements Engineering Conference (RE)*. IEEE, sep 2012.

[10] Jane Cleland-Huang, Brian Berenbach, Stephen Clark, Raffaella Settimi, and Eli Romanova. Best practices for automated traceability. *Computer*, 40(6):27–35, jun 2007.

[11] J.H. Hayes, A. Dekhtyar, Senthil Sundaram, and S. Howard. Helping analysts trace requirements: an objective look. pages 249– 259, 10 2004.

[12] Patrick Rempel and Patrick Mader. A quality model for the systematic assessment of requirements traceability. In *2015 IEEE 23rd International Requirements Engineering Conference (RE)*. IEEE, aug 2015.

[13] Patrick Mader, Orlena Gotel, and Ilka Philippow. Getting back to basics: Promoting the use of a traceability information model in practice. In *2009*

ICSE Workshop on Traceability in Emerging Forms of Software Engineering. IEEE, may 2009.

[14] Patrick Mader, Paul L. Jones, Yi Zhang, and Jane Cleland-Huang. Strategic traceability for safety-critical projects. *IEEE Software*, 30(3):58–66, may 2013.

[15] Patricia Lago, Henry Muccini, and Hans van Vliet. A scoped approach to traceability management. *Journal of Systems and Software*, 82(1):168–182, jan 2009.

[16] Hazeline U. Asun ion. Towards practical software traceability. In *Companion of the 30th International Conference on Software Engineering*, ICSE Companion '08, page 1023–1026, New York, NY, USA, 2008. Association for Computing Machinery.

[17] Achraf Ghabi and Alexander Egyed. Exploiting traceability uncertainty among artifacts and code. *Journal of Systems and Software*, 108:178–192, oct 2015.

[18] R. M. Parizi, S. P. Lee, and M. Dabbagh. Achievements and challenges in state-of-the-art software traceability between test and code artifacts. *IEEE Transactions on Reliability*, 63(4):913–926, 2014.

[19] G. Antoniol, G. Canfora, G. Casazza, A. De Lucia, and E. Merlo. Recovering traceability links between code and documentation. *IEEE Transactions on Software Engineering*, 28(10):970–983, 2002.

[20] M. Mirakhorli and J. Cleland-Huang. Detecting, tracing, and monitoring architectural tactics in code. *IEEE Transactions on Software Engineering*, 42(3):205–220, 2016.

[21] Patrick Rempel, Patrick Mäder, Tobias Kuschke, and Jane Cleland-Huang. Mind the gap: assessing the conformance of software traceability to relevant guidelines. In *Proceedings of the 36th International Conference on Software Engineering*. ACM, may 2014.

[22] Andrea De Lucia, Rocco Oliveto, and Genoveffa Tortora. Adams re-trace. In *2008 ACM/IEEE 30th International Conference on Software Engineering*, pages 839–842, 2008.

[23] Jun Lin, Chan Chou Lin, Jane Cleland-Huang, Raffaella Settimi, Joseph Amaya, Grace Bedford, Brian Berenbach, Oussama Ben Khadra, Chuan Duan, and Xuchang Zou. Poirot: A distributed tool supporting enterprise-wide automated traceability. In *14th IEEE International Requirements Engineering Conference (RE'06)*, pages 363–364, 2006.

[24] Jane Huffman Hayes, Alex Dekhtyar, and Senthil Karthikeyan Sundaram. Advancing candidate link generation for requirements tracing: The study of methods. *IEEE Transactions on Software Engineering*, 32(1):4, 2006.

[25] Mona Rahimi and Jane Cleland-Huang. Evolving software trace links between requirements and source code. In *2019 IEEE/ACM 10th International Symposium on Software and Systems Traceability (SST)*. IEEE, may 2019.

[26] Barbara Paech; Antje Knethen. A survey on tracing approaches in practice and research. *Fraunhofer IESE*, 7(095.01/E), January 2002.

[27] Muhammad Atif Javed and Uwe Zdun. A systematic literature review of traceability approaches between software architecture and source code. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering - EASE '14*. ACM Press, 2014.

[28] Ken Peffers, Tuure Tuunanen, Marcus A. Rothenberger, and Samir Chatterjee. A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3):45–77, dec 2007.

[29] Alan Hevner, Alan R, Salvatore March, Salvatore T, Park, Jinsoo Park, Ram, and Sudha. Design science in information systems research. *Management Information Systems Quarterly*, 28:75–, 03 2004.

[30] Anne Cleven, Philipp Gubler, and Kai M. Hüner. Design alternatives for the evaluation of design science research artifacts. In *Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology - DESRIST '09*. ACM Press, 2009.

[31] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, EASE'08, page 68–77, Swindon, GBR, 2008. BCS Learning Development Ltd.

[32] Ali Alsaawi. A critical review of qualitative interviews. *SSRN Electronic Journal*, 2014.

[33] Ilker Etikan. Comparison of convenience sampling and purposive sampling. *American Journal of Theoretical and Applied Statistics*, 5:1, 01 2016.

[34] Johnny Saldana. *The Coding Manual for Qualitative Researchers*. SAGE Publications Ltd, 4 edition, March 2021.

[35] Elizabeth J. Halcomb and Patricia M. Davidson. Is verbatim transcription of interview data always necessary? *Applied Nursing Research*, 19(1):38–42, 2006.

[36] Greg Guest; Kathleen M. MacQueen; Emily Namey;. *Applied Thematic Analysis*. SAGE Publications, Inc., 2012.

[37] Prabhakar Mishra, Chandra Pandey, Uttam Singh, and Anshul Gupta. Scales of measurement and presentation of statistical data. *Annals of Cardiac Anaesthesia*, 21:419–422, 10 2018.

[38] Per Runeson and Martin Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Softw. Engg.*, 14(2):131–164, 2009.

[39] C. Wohlin, Martin Höst, and Kennet Henningsson. Empirical research methods in software engineering. In *ESERNET*, 2003.

[40] M. Patton. Enhancing the quality and credibility of qualitative analysis. *Health services research*, 34 5 Pt 2:1189–208, 1999.

[41] Tina A Coffelt. Confidentiality and anonymity of participants. 2017.

[42] H. Gall, M. Jazayeri, and J. Krajewski. Cvs release history data for detecting logical couplings. In *Sixth International Workshop on Principles of Software Evolution, 2003. Proceedings.*, pages 13–23, 2003.

[43] J. Cleland-Huang, G. Zemont, and W. Lukasik. A heterogeneous solution for improving the return on investment of requirements traceability. In *Proceedings. 12th IEEE International Requirements Engineering Conference, 2004.*, pages 230–239, 2004.

[44] S. Maro. Addressing traceability challenges in the development of embedded systems. 2017.

[45] A. Agrawal, S. Khoshmanesh, M. Vierhauser, M. Rahimi, J. Cleland-Huang, and R. Lutz. Leveraging artifact trees to evolve and reuse safety cases. In *2019*

IEEE/ACM 41st International Conference on Software Engineering (ICSE), pages 1222–1233, 2019.

[46] Jonathan I. Maletic, E. Munson, A. Marcus, and T. N. Nguyen. Using a hypertext model for traceability link conformance analysis. 2003.

[47] Nicolas Anquetil, Uirá Kulesza, Ralf Mitschke, Ana Moreira, Jean-Claude Royer, Andreas Rummler, and André Sousa. A model-driven traceability framework for software product lines. *Software & Systems Modeling*, 9(4):427–451, jun 2009.

[48] Iflaah Salman, Ayse Tosun Misirli, and Natalia Juristo. Are students representatives of professionals in software engineering experiments? In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 1, pages 666–676, 2015.

A

Appendix 1 - Iteration 1

A.1 Interview Questions

These questions will be asked in in the interview:

1. Do you agree with the generic definition of X? (Y/N)
   (a) **If No:**
      i. How would you define this notion?
2. Can trace link correctness be evaluated with regards to X with the data available within the current dataset? (Y/N)
   (a) **If Yes:**
      i. What data exists to support this notion of correctness?
   (a) **If No:**
      i. Could the dataset be extended to support this notion of correctness on trace links in the project?
      i. **If Yes:**
         A. What data should be added?
3. Based on your experience, is this notion of correctness applicable on any project? (Y/N)
   (a) **If Yes:**
      i. What data should be added?
4. Based on your experience, on which model level would you apply this notion of trace link correctness?
5. Based on your experience, do you believe the definition of this notion of correctness can be generic or must be defined specifically for each project and why?

A.2 Interview Flow Chart

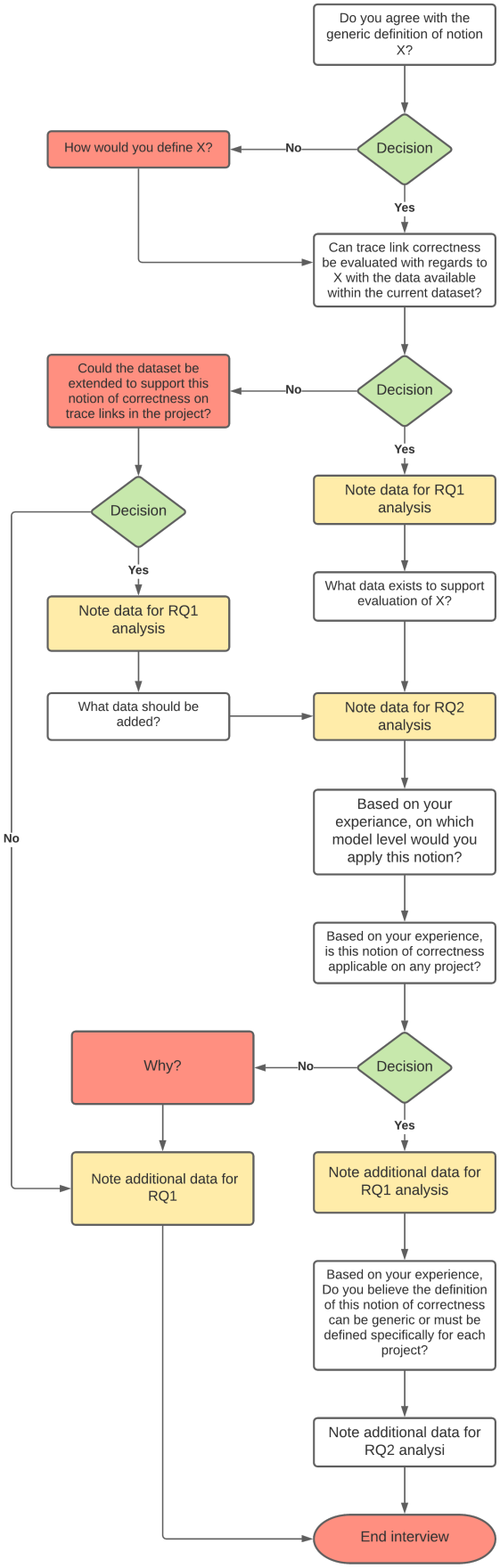

Figure A.1: Graphical representation of the interview flow

A.3 HIPAA and MobSTr Projects

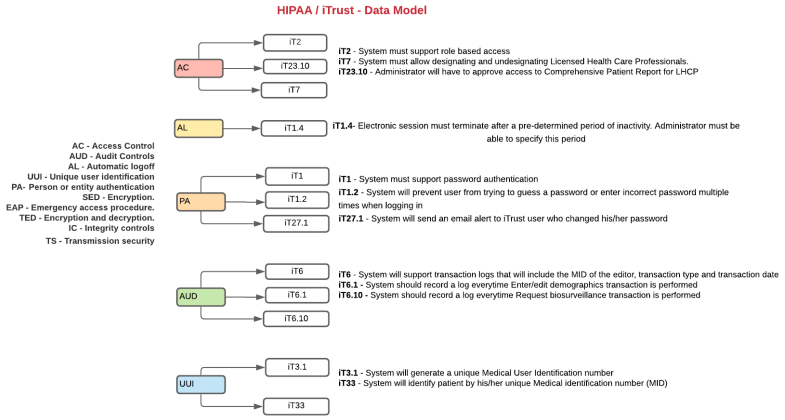

Figure A.2: Visual representation of iTrust dataset from HIPAA project used in the interview analysis

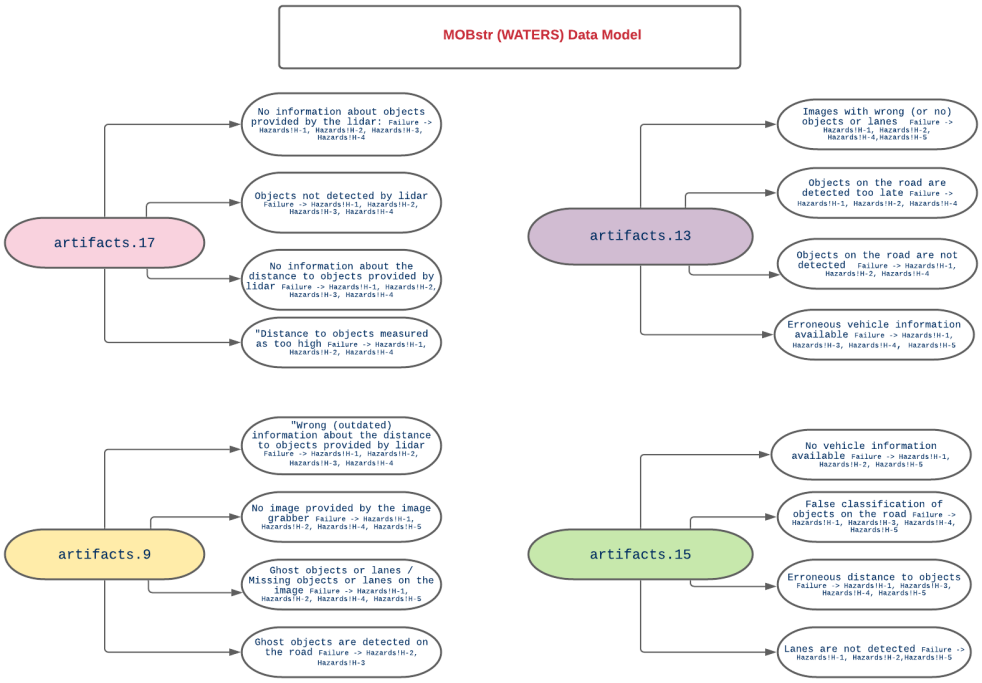

Figure A.3: Visual representation of MobSTr dataset from PANORAMA Research Project project used in the interview analysis

Table A.1: First Draft Notions of Trace Link Correctness

Level	Candidates	Concept in Literature	Extracted Generic Definition
Information Model and Data Model	Versioning	The maintainer must check if implications caused by evolving connected models arise. If a versioning solution exists, the traceability model must be appropriately updated with respect to the new versions of the models, i.e., one must decide if there should (still) be a link or not. Versioning emphasise the historical changes of the classes, it carries all the changes e.g.when new classes are added to the system and this classes are changed. In [15] the authors developed the premise that if artifacts of different types (e.g., src.cpp and help.doc) are co-changed with high frequency over multiple versions, then such artifacts potentially have traceability link between them.	Versioning is about maintaining a change history log and securing that information about the evolution of each artifact is stored. As part of the concept each change to any given artifact can be logged and this enables tracing changes to a certain point in time. Furthermore information about author of the change is frequently included and sometimes a complete interaction log showing what type of interaction was performed to each artifact in the system during development.
Information Model	Granularity	Trace link granularity must be clearly defined, in the TIM and RTMs and must be periodically evaluated in order to ensure that trace links are created at the correct granularity. As part of rationalization relations are expressed between traceable specification, a software specification with different level of granularity such as document, model, diagram, use case, etc. The solution suggested is that the granularity of the links should be defined explicitly in the traceability meta-model and the traceability links should be checked regularly to ensure that the links are created with the right level of granularity.Automated approaches tend to have difficulties working with various levels of granularity.The granularity describes the granularity of the entities involved (e.g., classes or attributes/methods of an object-oriented analysis, paragraphs or sentences of a textual requirements document)	Granularity is about relating a link to a specific level of an artifact. Each artifact can have various levels. Examples could be UML diagram, file, class, method and code as different levels for the same code artifact. Another example could be requirements document, specific requirement and specific step in a specific requirement as different levels when linking to requirement artifacts.
Information Model	Purpose	In order to minimize negative - return traces, it is important to evaluate WHY a link is being created, so that the most appropriate and useful type of link can be deployed.	Purpose is about securing that each permitted tracelink serves a specific purpose and thereby provides value. This means the effort to create and maintain the trace links brings return on invested resources.
Information Model and Data Model	Lifetime Lifespan	Differentiate between throw-away and long-term traces. Throw-away traces that are useful only during development, and those that should be maintained in the long-term.	Lifetime/Lifespan is about setting an end date to each trace link where maintenance will no longer be performed.

		The semantics must thus be adapted and updated continually during the lifetime of the project, not only by adding new "types" of links, but also by refining and even deactivating existing types.	
Data Model	Non-duplicated Tracelinks	Prevent duplicated links by storing them in a database-like repository. Either define constraints that prevent redundant links from being created or regularly execute trace queries to find duplicated links and remove them.	Non-duplicated trace links is about securing that there is no duplicated information in trace link data. This in order to avoid future update errors where only one link has been updated but not other duplicates. This makes it impossible to know which one is correct and which link was not updated.correctly at some point in the project life-cycle.
Information Model and Data Model	Unique Artifact Identification	A fundamental principle of traceability is that each traceable artifact must have a unique identifiers. Furthermore, prefixes used to distinguish artifact types should be unique across the project as well as intuitive to stakeholders.Artifacts need to be unique which is a characteristic of a traceable artifact.	Unique ID is about being able to identify a specific artifact through it's ID. The ID only belongs to one single artifact in the project, thereby identifying it uniquely via the ID.
Data Model	Mandatory Tracelinks	Traceability between artifact types that is required by a guideline can only be considered complete if at the project level every single artifact of the requested source artifact type is traced directly via a tracelink or transitively via a trace path to an artifact of the requested target artifact type.	Mandatory trace links is about securing that all trace links that must exist according to used guidelines or regulation are created. This secures that applicable guidelines and regulation are fulfilled by the project.
Information Model	Arity	Determine the number of relations	The arity of a link specifies the number of its end points.

B

Appendix 2 - Iteration 2

B.1 Questionnaire Questions

Following are questions in the questionnaire:

1. Do you agree with the generic definition of X? (Y/N)
   (a) **If No:**
      i. How would you define this notion?
2. Do you think that notion X exists in the dataset in the given example above? (Y/N)
   (a) **If Yes:**
      i. What existing properties/data would you use in order to identify notion X?
   (a) **If No:**
      i. Could the dataset in the example be extended in order to identify notion X?
      i. **If Yes:**
         A. What data should be added and why?