Requirements Engineering
https://doi.org/10.1007/s00766-023-00412-z

ORIGINAL ARTICLE

A systematic literature review of pre-requirements specification traceability

Julia Mucha[1] · Andreas Kaufmann[1] · Dirk Riehle[1]

Received: 4 November 2022 / Accepted: 20 December 2023

Abstract
Requirements traceability (RT) is the ability to link requirements to other software development artifacts. In pre-requirements (pre-RS) traceability, requirements are linked to their origin, such as interviews with stakeholders, meeting protocols, or legacy systems. Compared with post-RS traceability, which links requirements to source code and other later artifacts, pre-RS traceability has seen much less research. This article presents a systematic literature review of pre-RS traceability based on 77 articles published between 1992 and 2022, aiming to provide a comprehensive overview of its use cases, benefits, problems, and solutions. Through the analysis of existing literature, this review identifies gaps for future research and establishes a foundation for future investigations in the field of pre-RS traceability.

Keywords Pre-requirements specification traceability · Requirements traceability · Requirements engineering · Systematic literature review

1 Introduction

Keeping requirements up-to-date and understanding their context is crucial for project success, especially if the project grows and becomes more complex. A comprehensive understanding of the domain and thorough process documentation are required to answer questions like, *what is the reason behind a particular requirement? Why was this requirement changed? Who was involved?*. Requirements traceability and in particular pre-requirements specification (pre-RS) traceability can answer those questions, provided that such traceability is integrated into the engineering process.

Pre-RS traceability "[...] refers to those aspects of a requirement's life prior to inclusion in the RS"[37] by so-called trace links. Post-requirements specification (post-RS) traceability is the ability to link requirements to artifacts based on the RS. While post-RS traceability has received more attention in previous research [1, 4, 38, 100, 108], the potential of pre-RS traceability for cost and quality improvements in software development has been recognized [17, 36]. However, the systematic exploration of pre-RS traceability in the research literature has been limited.

To establish pre-RS traceability, engineers document stakeholder information relevant to requirements creation from the project's inception to subsequent refinements. Stakeholder information includes interviews with stakeholders, meeting protocols, documentation of decisions made, etc. These documentations are typically in textual form, but hypermedia such as audio recordings, photos, and videos can also be utilized. This makes pre-RS traceability a more dynamic procedure than post-RS traceability because it has to handle initially unstructured qualitative and varying data of different types and formats [98]. The management and maintenance of trace links face similar challenges to post-RS traceability, such as the detection of obsolete links.

The diverse needs of practitioners and their various roles within a project are often considered crucial for determining an appropriate traceability strategy [34, 60, 75, 89]. The perceived effort and workload associated with pre-RS traceability are often seen as too high [37, 100, 107]. As a result, pre-RS traceability is commonly applied only to fulfill regulatory compliance for safety-critical systems, without

✉ Julia Mucha
mail@juliamucha.de

Andreas Kaufmann
andreas.kaufmann@fau.de

Dirk Riehle
dirk@riehle.org

1 Department of Computer Science, Friedrich-Alexander-University Erlangen Nürnberg, Martensstr. 3, 91058 Erlangen, Germany

fully realizing the potential benefits, such as a transparent RS creation process that encourages exchange and avoids conflicts during requirement discussions.

Although the challenges and benefits of pre-RS traceability are unique and significant, pre-RS traceability is commonly treated only as an afterthought to post-RS traceability.

In this article, we document the state-of-the-art of pre-RS traceability by analyzing current problems, use cases, techniques, and tools. We conducted a systematic literature review (SLR) of articles published in 14 academic journals and over 15 conference proceedings from January 1992 until June 2022. As part of the review process, we performed qualitative data analysis (QDA) on the 77 papers we had retrieved as relevant to our topic. The contributions of this article are:

- Systematic summary of problems and solutions
- Identification of influencing factors for pre-RS traceability
- Systematic summary of techniques for managing pre-RS traceability and the extent to which they have already been evaluated
- Topics for future research on pre-RS traceability

The article is structured as follows. First, we present related work in Sect. 2. In Sect. 3, we present the research questions, before we elaborate on the review method in Sect. 4. Afterward, we present the results of the SLR in Sect. 5. The results section starts with a quantitative literature review and continues to present the qualitative results. Here, we answer the research questions. We close the article with a discussion in Sect. 6, a section about threats to validity 7, and a conclusion in Sect. 8.

2 Related work

Research has extensively focused on requirements traceability. Gotel and Finkelstein [37] defined the destination between pre-RS traceability and post-RS traceability. Based on their empirical study, they suggest more attention is needed in the area of pre-RS traceability because it has a significant influence on a project's success.

Researchers have conducted literature reviews on requirements traceability focusing on post-RS traceability, contributing much state-of-the-art knowledge [14, 21]. Further research highlights and uses the benefits of using trace links to support engineering and maintenance of software systems [24, 45, 87, 90, 103]. They are predominantly focused on post-RS traceability. However, several literature surveys in the field of requirements traceability have stated that not much has been done in the area of pre-RS traceability [1, 3, 6, 8, 29, 52, 66, 82, 99, 100]. Most of these studies have highlighted the need for more attention to pre-RS traceability. In line with these suggestions, we conducted a systematic literature review to contribute to the state-of-the-art knowledge on pre-RS traceability and provide a foundation for future research.

Pre-RS traceability can benefit from the findings related to post-RS traceability and requirements traceability, for example in the *definition*, *production*, and *extraction* of trace links presented by [75]. We considered these aspects and different techniques for realizing pre- and post-RS traceability in Sect. 5.5. Additionally, strategies for realizing requirements traceability need to consider the characteristics of their respective environments.

This observation was made by Ahmad and Ghazali [1], who identified problems of requirements traceability within small projects and developed guidelines for documenting trace information. They found that taking into account the characteristics of the project and the company is important in developing a suitable traceability strategy.

Considering the environment also entails considering the individuals involved in traceability activities. This was recognized by previous research [32, 65, 78, 79, 88]. We explored the influencing characteristics and individuals involved in the project environment, which influence not only post-RS but also pre-RS traceability, as discussed in 5.3.

However, there are significant differences between pre-RS traceability and post-RS traceability. While post-RS traceability tends to link formal and structured artifacts such as requirements, models, source code, and tests, pre-RS traceability often needs to link informal and unstructured information such as protocols, interview transcripts, notes, or figures [98]. This information is necessary to understand the problem domain and derive requirements. Kaindl [48, 49] developed an approach including tool-support named RETH (Requirements Engineering Through Hypertext). RETH supports the linking of non-text media to requirements.

Automated approaches that rely on natural language processing or artificial intelligence face unique obstacles when trying to process various source types [2, 17]. These approaches have mainly concentrated on post-RS traceability, which involves processing formal and structured sources [13, 26] and creating RS based on software engineering models [68]. In our study, we considered (semi-) automated approaches if they addressed pre-RS traceability, as discussed in 5.5.

One problem is that the benefits of pre-RS traceability are not clear to all individuals involved. Altaf et al. [4] try to address this issue by developing a visualization for pre-RS traceability. attempted to address this issue by developing a visualization for pre-RS traceability. We built upon their findings, as they also published a list of benefits of pre-RS traceability. We have already investigated the issue of unclear benefits through a systematic literature review extended by a qualitative survey, and our work has been published [84]. This paper details the methodology, presents all findings of the systematic literature review, which includes the findings related to use cases and benefits, and adds further literature about pre-RS traceability published between 2020 and 2022.

Pre-RS traceability is also relevant in agile projects [19, 44, 95]. In agile projects, user stories are commonly used to document requirements that need to be implemented. However, user stories are typically not intended for long-term documentation. As a result, adaptations to software functionality often do not lead to an update of the corresponding existing user story but rather the creation of a new user story describing the adaptation. Therefore, we excluded the traceability between customer statements and user stories, as this requires separate research.

3 Research questions

This SLR broadly assesses and reviews the state-of-the-art of pre-RS traceability. We answer three key research questions, covering use cases and benefits, problems and solutions, and current techniques of pre-RS traceability.

(RQ1) What are the use cases and benefits of pre-RS traceability?

During a preliminary literature analysis, we found that the assumed benefits of pre-RS traceability vary and are often not clear. The same holds for use cases of pre-RS traceability. Therefore, we asked RQ1 to take a deeper look at the relevance of the topic. We analyzed common use cases, their benefits, and their motivations.

(RQ2) What are problems and solutions of pre-RS traceability?

The analysis of current problems and existing solutions is an important part of the state-of-the-art and creates a base for future research. Furthermore, answering this question helps us identify unsolved problems and therefore direct future research.

(RQ3) What are pre-RS trace techniques?

We also asked to identify current techniques (including tools, methods, etc.) and how they help to solve the challenge of realizing pre-RS traceability. Correlating techniques with problems and solutions allowed us to identify missing techniques in general and missing evaluations of existing techniques in particular.

4 Review method

We followed the procedure described by Kitchenham for the systematic literature review (SLR) [58]. In the beginning, a research protocol was drawn up to describe the background and to define all important cornerstones like research questions, research process (Sect. 4.1), search strategy (Sect. 4.2), selection criteria (Sect. 4.3), data extraction (Sect. 4.4), data synthesis (Sect. 4.5), and a work program consisting of work packages. The protocol provides a full audit trail of our research. Regular peer debriefings (Sect. 4.6) were conducted to ensure high quality during the procedure and a good fit of research questions to research design [93].

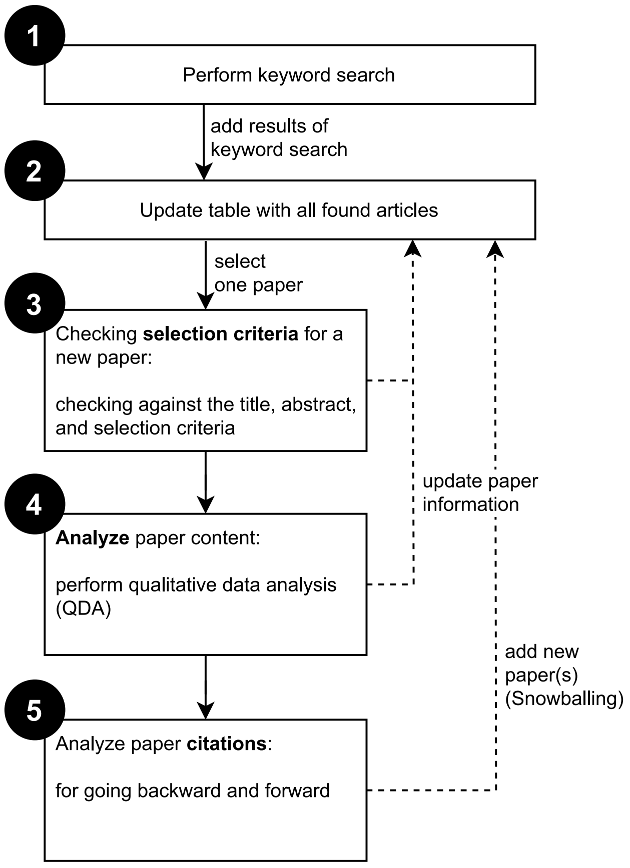
1
Perform keyword search
add results of keyword search
2
Update table with all found articles
select one paper
3
Checking **selection criteria** for a new paper:
checking against the title, abstract, and selection criteria
update paper information
4
Analyze paper content:
perform qualitative data analysis (QDA)
add new paper(s) (Snowballing)
5
Analyze paper **citations**:
for going backward and forward

Fig. 1 Visualization of the data extraction and synthesis

4.1 Research process

Figure 1 visualizes the iterative research process. In the *first step*, we performed a keyword search and obtained results. These results were then added to the literature table during the *second step*. The *third step* involved checking the articles against the selection criteria. If an article was identified as relevant, we proceeded to the *fourth step*, which involved data synthesis. The *fifth step* consisted of analyzing the citations in order to find additional relevant articles.

Throughout each step, we continuously updated the table of articles with information about each paper or any new papers discovered through the citation analysis (snowballing).

Table 1 Search terms and results per database

Search term	Google scholar	IEEE xplorer	ACM	Web of science
“Pre-requirements specification traceability”	62	7	1	1
“Pre-requirements specification”	133	20	6	1
“Requirement provenance”	42	1	2	11

4.2 Search strategy

To find relevant articles we used two different search strategies. We first performed a pilot keyword search. Based on the topic and research questions, the keyword “traceability” is the most obvious, but also the most general. The set of resulting articles included many papers that focus on post-RS traceability, as well as papers that fall outside of the scope of our work. We evaluated making the search query more specific by searching for “traceability AND requirements engineering". This, however, did not improve the results substantially. We found in our analysis of the results that the vast majority of relevant results explicitly used the term “pre-RS traceability” in the text. Some relevant papers also used “requirement provenance” or “pre-requirements specification". We therefore decided on using these three individual search terms that cover different designations of pre-RS traceability. Table 1 shows the keywords and the associated number of articles found in each database. We then carried out snowballing by performing forward and backward search to not miss relevant articles independently of the vocabulary used in the text [105].

4.3 Selection criteria

To identify relevant articles, we defined the following questions to address the selection criteria:

1. Is the article written in English?
2. Is the article peer-reviewed?
3. Is the article not yet in the collection of relevant articles? (Remove duplicates)
4. Is the article about a technique or method to link RS with their source artifacts?
5. Is the article about an overview that presents different techniques, issues, and/or problems to link source artifacts with RS?
6. Is the article about an evaluation of a technique or method which links source artifacts with RS?

For the inclusion of an article, questions 1, 2, and 3 must be answered in the affirmative and at least one of the other questions 4 to 6 should be answered with “yes", too.

4.4 Data extraction

To keep track of the articles and their state of analysis, we created and continuously updated the table of articles. The table is presented as document [73]. This table consists of the following columns:

- **Type** represents the search strategy with which the article was found (K = keyword search, B = backward search, F = forward search).
- **Resource** represents the source of the found article. This can be a specific database (Google Scholar, etc.) or an article that was the basis for the forward or backward search.
- **Search Term** by which we found the article.
- **State** of the analysis and the decision (to do, reviewed, denied, no access).
- **Type of Content** (Tech = technique or method, Sum = summery or survey, SuTe = summary followed by developed technique, EmSt = empirical study, CSt = case study or evaluation, Gly = glossary)
- **No. of Citations** represents the popularity of the article.
- **Authors** of the article.
- **Title** of the article.
- **Year** of publication.
- **Outlet** contains the name of the journal, conference, or workshop.
- **Analysis Date** of when we finished the analysis.

Table 2 summarizes the number of articles in the different states. After applying the selection criteria, we identified 41 articles by keyword search and 36 articles by backward and forward search. Finally, we included 77 articles (list in Appendix A) in our study about pre-RS traceability from the whole field of RE.

4.5 Data synthesis

Based on Kitchenham [58] we started with a quantitative synthesis to provide an overview of the characteristics of the article in Sect. 5.1. However, the main focus was on the descriptive (non-quantitative) synthesis by performing QDA to answer the research questions.

Table 2 No. of articles per state of selection

Selection state	No. of articles
Based on keyword search (sum of found articles based on Table 1)	287
Included articles from keyword	41
Included articles from forward and backward search along with referenced articles	36
Included articles total	77

4.5.1 Quantitative analysis

To get an idea of the scope of the research topic, we analyzed the number of publications per year and outlet. In addition, we analyzed the types of content, whether the paper introduces a new technology, covers a case study, or presents another empirical study. This gives us insight into what the focus is in this research area. The results are presented in Sect. 5.1.

4.5.2 Qualitative data analysis (QDA)

The articles identified as relevant were analyzed using QDA to extract RQ-related information from qualitative data. To perform QDA, we applied the iterative coding process of Corbin and Strauss [16]. Each iteration consisted of coding one document based on the three steps: open coding, axial coding, and selective coding. **Open coding** means annotating relevant segments of text with codes. The codes represent concepts. In the step of **axial coding** these concepts, called codes, have been structured in a hierarchical code system by grouping similar concepts into categories. The final step of an iteration, **selective coding**, means arranging all codes and relating them to the central phenomenon of the study, called the core category, thus refining the focus on the research questions. The core category in our study is the research field of pre-RS traceability, focusing on use cases, challenges, solutions, and techniques for implementation.

Based on a randomly selected pilot sample, we developed and continuously refined a basic code system. Table 3 shows the main categories of the hierarchical code system and the number of assigned codings sorted by related research question. The first two categories are not assigned to one of the three research questions. However, these categories emerged during the analysis to define and thus delimit terms. This created the common knowledge base on which we will answer the research questions.

Figure 2 presents the number of codes and codings by iteration. The red line flattened from the 63rd iteration whereas the blue line continues to rise. This means there are only minimal changes to the code system by adding or deleting codes, but codes were continuously assigned to segments of text. This is an indication of saturation because little new information was emerging that was changing the code system.

Table 3 Main Codes of the code system, related to the research questions, and the number of codings

Code related to research question	No. of codings
Definition of terms (Sect. 5.2)	
Requirements Traceability (RT)	5
Pre-RS traceability general (+ sub codes)	81
RQ1: use cases and benefits (Sect. 5.3)	
Users applying pre-RS traceability (+ sub codes)	31
Use cases and benefits (+ sub codes)	249
RQ2: problems and solutions (Sect. 5.4)	
Problems and challenges (+ sub codes)	275
Consequences of poor pre-RS traceability	5
Solutions and suggestions (+ sub codes)	258
RQ3: techniques (incl. tools) (Sect. 5.5)	
Trace techniques (+ sub codes)	212
Trace tools (+ sub codes)	48

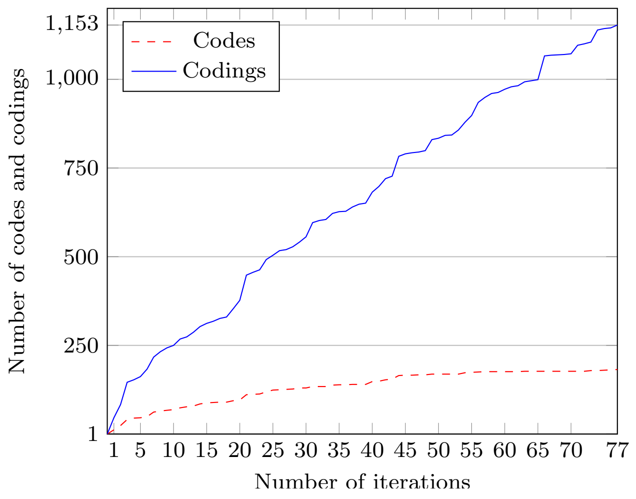

Fig. 2 Number of codes (concepts) and codings (assignments) per iteration

4.6 Research quality assessment

In addition to a continuous professional exchange between all co-authors, we carried out regular peer debriefings to ensure high quality of protocol and execution during the procedure and a good fit of research questions to research design. Based on Spall [93]: *"Peer debriefing contributes to*

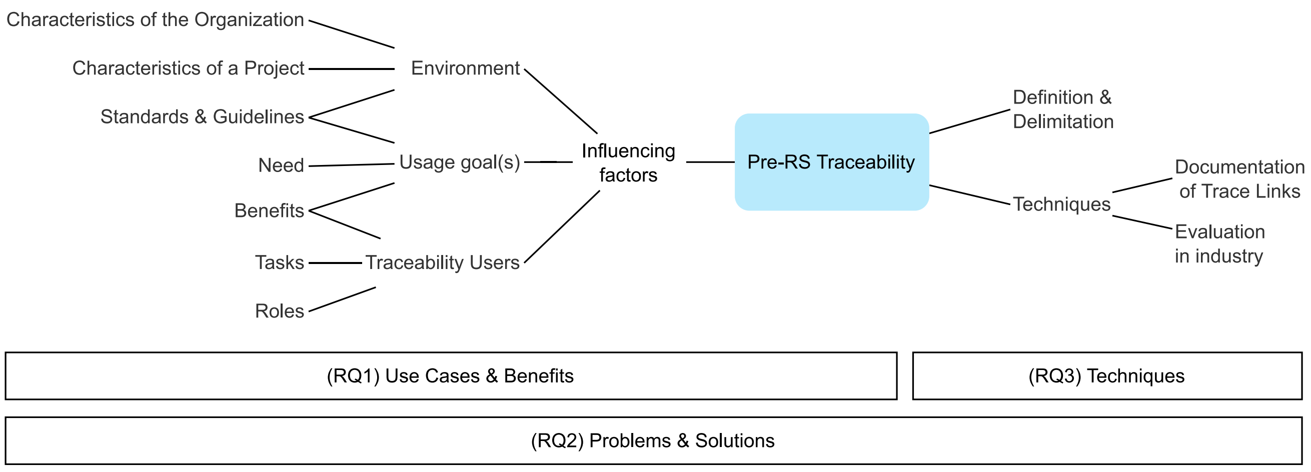

Fig. 3 Topics emerged from the codes including the scope of the research questions

confirming that the findings and the interpretations are worthy, honest, and believable." As a debriefer an experienced researcher in our group was selected since he was available for us and had sufficient domain knowledge. During these peer-debriefings, we reviewed and evaluated the current state of our code system to critically review the hierarchical structure as well as the naming and definition of the codes. This also allowed us to identify and resolve inconsistencies and improve the reliability of the analysis. The peer debriefings also helped us identify meaningful overlaps of different concepts. All identified issues were discussed jointly to decide how to proceed.

We performed three peer debriefings, one after the pilot sample of 10 articles and one after the 25th iteration. The third peer debriefing was conducted during the last iterations of the data synthesis. Each debriefing was documented in a peer debriefing protocol which consisted of the following information:

- **Date and title**
- **Participants** consisting of the name of the debriefer and the one who carried out the study
- **Improvements from last peer debriefing** if existing to discuss these improvements
- **Current state, method, and workflow** to bring the debriefer up to date and to focus on current challenges to be discussed
- **Improvements** containing future actions to be executed

5 Results

In the following, we first present the results of the quantitative and meta-data analysis (Sect. 5.1) and key terms (Sect. 5.2) to reach a common understanding of pre-RS traceability, before we answer our research questions in Sects. 5.3 to 5.5.

Figure 3 provides an overview of important topics that emerged during our analysis through the codes created. The mapping of the topics to the research questions is derived from the mapping of the codes to the research questions (Table 3). The topics of environment, the usage goals, and the users of trace links are strongly linked to RQ1 by the main codes *use cases and benefits*, *Users applying pre-RS traceability*, and their subcodes. However, problems with associated solutions (RQ2) occur across all codes of the code system. Some topics have already been explored more, others less. A lot of research is being done to develop different techniques for realizing pre-RS traceability (RQ3), but there is a lack of evaluation inside industry projects (Sect. 5.5).

The decision to use a particular technique to realize traceability in a project is important. Therefore it is necessary to identify influencing factors and characteristics to support the decision. The RQ1 on use cases and benefits provides some basic information in this area (Sect. 5.3).

5.1 Quantitative and meta-data analysis

From 1992 to 2022, 77 relevant research articles were published. Figure 4 shows the number of articles published per year. We identified 52 relevant conference articles and 16 relevant journal articles. Nine relevant documents came from books, reports, and workshops. More than half of the articles (52%) came from the *International Requirements Engineering Conference (RE)* and 16% from *Traceability in Emerging Forms of Software Engineering (TEFSE)*. The 16 journal articles originate from 14 different academic journals. A large number of different sources demonstrates the need for this topic in different areas and thus also for systematic processing by an SLR.

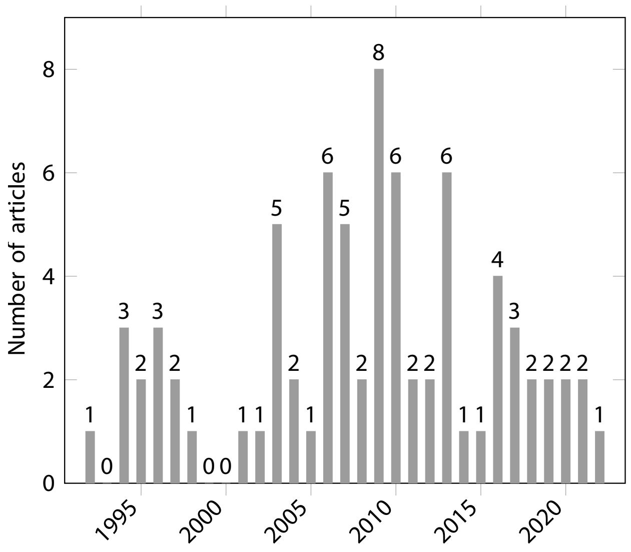

Fig. 4 The number of articles published per year

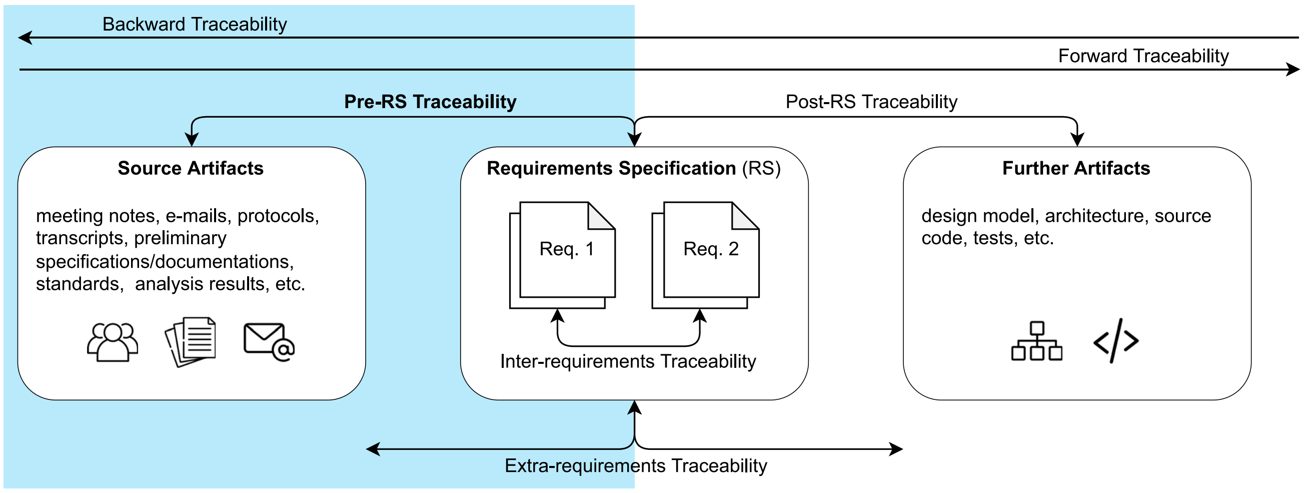
Backward Traceability
Forward Traceability
Pre-RS Traceability
Post-RS Traceability
Source Artifacts
meeting notes, e-mails, protocols, transcripts, preliminary specifications/documentations, standards, analysis results, etc.
Requirements Specification (RS)
Req. 1
Req. 2
Inter-requirements Traceability
Further Artifacts
design model, architecture, source code, tests, etc.
Extra-requirements Traceability

Fig. 5 Summary of the trace types

The most common focus of publications was on novel techniques. Case studies or other empirical studies got less attention.

5.2 Definition of terms

We identified diverging terminologies in the literature. A common understanding of the terminology is necessary for later use and to avoid misunderstandings. Figure 5 presents the different types of traceability.

A **requirements specification** (RS) contains all requirements (functions, performance, design constraints, and other attributes) captured and maintained by requirements engineers or analysts at the beginning and during a project. It is usually written in natural language to be accessible to members of a project such as stakeholders, suppliers, etc. [31, 51]. Furthermore, a "[...] software requirement specification is traceable if (1) the origin of each of its requirements is clear and if (2) it facilitates the referencing of each requirement in future development or enhancements documentation." [96]. Point (1) refers to pre-RS traceability and (2) to post-RS traceability.

Pre-requirements specification (pre-RS) traceability "[...] refers to those aspects of a requirement's life prior to inclusion in the RS" [37]. Most of the articles refer to this definition. Sometimes pre-RS tracing is also called **upstream tracing** [86]. In contrast, **post-RS traceability** is the ability to trace between requirements and artifacts such as source code, tests, etc. that are based on them.

Forward and **backward traceability** describe the ability of relative tracing depending on the starting point. In the case of pre-RS traceability, forward tracing from a source artifact to a requirement "demonstrates how (and which) requirements in the RS satisfy individual needs" [36]. In contrast, backward traceability is realized by trace links referring back to the source of a requirement [1, 10, 75, 100]. **Inter-requirements traceability** is the ability to trace the dependencies between requirements [75]. **Extra-requirements traceability** is realized by trace links between requirements and other artifacts. As the description already shows, the different types can overlap [75].

In this way, pre-RS traceability and inter-requirements traceability overlap when requirements are refined or requirements are revised [22, 34, 75, 100].

Table 4 Different traceability user categories and their characteristics

Practice → Function ↓	Low-end user [78]	High-end user [78]
Provider [37]	*Low-end provider*: Creates trace links that he needs to create Doesn't know the usage goals of trace links Doesn't know benefits of created trace links	*High-end provider*: Creates good links Knows the usage goals of the trace links Knows benefits of the created trace links
End user [36, 37]	*Low-end end user*: Uses trace links minimally Doesn't see all benefits	*High-end end user*: Uses the links a lot Knows benefits Knows how to use trace links for improvements

5.3 (RQ1) What are the use cases and benefits of pre-RS traceability?

The analysis of the use cases shows how important it is to consider the involved people, the characteristics of a project (like size and longevity), the project's environment, and the usage goals of pre-RS trace links to decide on a particular and appropriate strategy to realize traceability. Therefore, this section starts by presenting different types of users (Sect. 5.3.1) and afterward presents use cases and benefits (Sect. 5.3.2).

5.3.1 Users applying pre-RS traceability

Goguen [32] identified three social groups: the client organization, the requirements team, and the development team. Our research focuses on the first two groups because they are mainly involved in pre-RS traceability.

Gotel and Finkelstein [37] as well as Ramesh [78] defined common user categories. For example, Gotel and Finkelstein [36, 37] introduce a distinction between **providers** and **end users**. Providers are people who can create trace links. End users are people who need information provided by trace links to do their jobs, but typically cannot create trace links. Distributing capability and benefits among different people can lead to a problem if the people do not have a common understanding. On one hand, providers can create trace links but often do not know the intended use of these particular trace links. On the other hand, end users do not know how to create trace links, but they know the intended use. It is a common problem in post-RS traceability, where providers usually are developers, and end users are requirements engineers. However, the same problem of distributing capabilities and benefits also occurs in pre-RS traceability when many different analysts, requirements engineers, and stakeholders are involved. Especially client organizations are typically end users [72]. Client organizations like to know which needs are already satisfied. This information can be made available using trace links between requirements and their source artifacts.

Ramesh [78] distinguishes between **low-end** and **high-end users**. "Low-end users view traceability simply as a mandate from project sponsors, whereas high-end users view traceability as an important component of a quality systems engineering process" [78].

Like Ramesh [78], Mäder et al. [65] made a more detailed distinction between users applying traceability based on the underlying motivation and practice.

Further categorizations are made by differentiating according to function. Therefore, Gotel and Finkelstein introduced the roles of the *principal agent*, *author agent*, and *documentor agent* in the context of the *contribution structure* [34].

However, low-end versus high-end users differ depending on the traceability practice and motivation, while providers versus end users differ in function and capabilities. The combination of these characteristics: traceability practice, motivation, function, and capabilities allows for more fine-grained categories. Table 4 defines and relates these user categories to each other. The table uncovers a conflict between *low-end providers* and *high-end end users* if they are part of the same project. High-end end users need much more information provided by trace links than low-end provider creates.

Considering the human factor, individual roles, and tasks inside a project is key to deciding on a strategy that realizes pre-RS traceability and appropriate tool support. Therefore, further investigations are needed on how to customize the strategy to the tasks and the roles.

5.3.2 Use cases and benefits

Tables 5 and 6 present all identified use cases and benefits ordered by descending number of codings from top to bottom.

Finding the source of a requirement to support understanding of the requirement's content and its context was most frequently mentioned as a use case for pre-RS traceability [8, 20, 47, 110] followed by the identification of responsible people [1, 4, 34]. Both use cases address the problem of "black-box" requirements [34] and clarify ambiguous or unclearly described requirements. The fulfillment of regulatory compliance or norms is a typical use case for traceability of safety-critical systems, but not the most frequently mentioned one [4, 65, 89, 100]. Especially low-end users mentioned this use case [78].

Table 5 Rankings of use cases of pre-RS traceability

Ranking of *use case* codes	No. of codings
1. Finding source (support understanding)	40
2. Responsibility identification	18
2. Fulfillment of regulatory compliance/norms	18
3. Impact analysis as part of change-management	17
4. Prove fulfillment of stakeholder needs	16
5. Support decision making as part of change-management	15
6. Getting the state of the RS	11
6. Manage system evolution (Maintaining)	11
6. Keeping track of history/relationships	11
7. Change-management in general	9
8. Knowledge management system	7
9. Requirements negotiations	5
9. Requirements prioritization	5

Table 6 Rankings of benefits of pre-RS traceability

Ranking of *benefit* codes	No. of codings
1. Monitor and gain knowledge for future projects	21
2. Improve product/software quality	13
3. Support reusability of requirements	10
4. Improve communication and collaboration	8
5. Reveal tacit knowledge	4
5. Satisfaction of stakeholders	4
6. Finding missing requirements	2
6. Reduction of maintenance costs	2
6. Finding unnecessary requirements	2

Traceability supports change management. Ravichandar et al. [81] describe traceability as "the cornerstone of change-management". This includes pre-RS traceability, by tracing backward from requirements to their source. The source provides valuable background information, such as previous decisions and the persons involved. The knowledge about the history of a requirement's creation and evolution can support future decisions [57, 89, 108] and contributes to a better understanding of the impact of subsequent requirement changes to realize impact analysis [1, 8, 107].

Monitoring RE artifacts based on the information provided by pre-RS traceability, and deriving knowledge for future projects are the most frequently mentioned benefits of pre-RS traceability [4, 30, 78, 92, 108]. These benefits are strongly related to the identification of reusable requirements and best practices to save costs and improve processes.

To analyze the relationship between use cases and benefits, we extracted and analyzed overlapping codings. This reveals three benefits that are not related to specific use cases: revealing tacit knowledge [77, 97, 98], finding missing requirements, and finding unnecessary requirements [107]. Tacit knowledge plays an important role in creating and maintaining an RS, as stakeholders and requirements engineers often have more knowledge than they communicate. Tacit knowledge of requirements engineers flows into the RS, but the tacit knowledge of the stakeholder often remains hidden. Pre-RS traceability can help discover requirements without linked source artifacts by backward tracing. These requirements can be based on tacit knowledge [77, 97, 98], or are no longer needed [107]. In contrast, finding missing requirements is possible by forward tracing based on the stakeholder needs [107]. If a need cannot be traced to specific requirements, the corresponding requirements may be missing.

Use cases of pre-RS traceability such as change-management, history tracking, system maintenance, knowledge management, etc. become more important the larger and more complex the project gets. However, smaller projects can benefit from pre-RS traceability as well. Ahmad and Ghazali [1] show this in a study of small projects.

More investigations are needed to

- identify relevant characteristics of a project (like size, longevity, or type of process) that influence practices used for realizing traceability,
- evaluate relationships between use cases and particular types of users applying traceability, and
- identify return on investment of pre-RS traceability.

5.4 (RQ2) What are problems and solutions of pre-RS traceability?

Gotel [36] called the "traceability problem" multifaceted with many different underlying problems. Providing reliable trace information depends on many, and frequently changing, factors related to a project environment. The "traceability problem" includes pre-RS traceability, which we analyzed more closely to collect underlying problems and existing solutions in the academic literature.

Tables 7 and 8 present the rankings of the *problem and challenge* and *solution and suggestion* codes. The ranking is based on the number of codings, starting with the most frequently coded code.

5.4.1 Problems and challenges

Many traceability problems are people-related. This was already recognized during early research and still is an area with great research potential today [33, 61, 101]. The most frequently mentioned people-related problem is to satisfy all involved roles, their interests, and their knowledge [36,

Table 7 Rankings of problems of pre-RS traceability

Ranking of *problem and challenge* codes	No. of codings
1. people-related problems (incl. 6 sub codes)	73
2. Inadequate documentation of trace information (incl. 4 sub codes)	34
3. No/poor maintenance of trace information (evaluation) (incl. 1 sub code)	27
4. No generalization/standards	24
4. Organizational problems (incl. 2 sub codes)	24
5. Tooling problems (incl. 2 sub codes)	18
6. No/poor identification of trace artifacts (incl. 1 sub code)	13
7. No/poor (semi-)automation	12
8. Bad access/presentation of trace information	11
9. Organize unstructured information	10
10. Missing usage goal for traces (incl. 1 sub code)	8
11. No/inadequate support managing large/distributed amount of data	5
11. Poor adaptability for project-specific needs	5
12. No/poor versioning support of traces	4
13. No/poor reusability	2

Table 8 Rankings of solutions of pre-RS traceability

Ranking of *solutions and suggestions* codes	No. of codings
1. Solutions regarding trace link specifications (incl. 4 sub codes)	60
2. people-related solutions (incl. 10 sub codes)	50
3. Obtain and record trace information (incl. 7 sub codes)	37
4. Enable valuable visualizations and presentations	18
5. Provide flexible tools/support	17
6. Create/improve the structure of RS (incl. 4 sub codes)	16
7. Improve generic trace model (incl. 2 sub codes)	15
8. Provide fitting storage for traces and data (incl. 1 sub code)	13
9. Organize individual (origin) artifacts	10
10. Establish templates and guidelines	8
11. Maintain trace links and artifacts (incl. 3 sub codes)	7
12. Configurable traceability strategies	3
13. Analyze and consider special attributes	2
14. Support information transfer	1

37, 61, 70, 72, 76, 78–81, 88, 94, 101, 108]. This applies to all the different types of users previously described in Sect. 5.3.2. The second most frequently mentioned people-related problem is that trace activities are seen as too much work, compared to the seen benefits [4, 25, 36, 37, 42, 62, 78, 81, 94, 100, 107, 108]. Often the benefits are not known to all people involved or the effort and benefits belong to different roles. This kind of distribution appears, for example, between providers and end users.

Many of the problems and challenges are related to each other. Therefore, we analyzed the overlap between codings to reveal significant co-occurrences. One strong correlation was identified between the inadequate maintenance of trace information and people-related problems. Maintenance tasks are often seen as time-consuming activities [4, 62]. Furthermore, it becomes more difficult to maintain requirements and trace data if there is no information about a responsible person or at least persons involved linked to these particular requirements [33, 40]. Losing source artifacts of requirements over time leads to inadequate documentation of trace information and vice versa. The time-consuming nature of maintenance tasks could potentially be reduced through automation. However, even if these tactics are employed, the lack of error-free (semi-) automation to create trace information leads to a lot of manual rework by checking the suggested candidate trace links [42, 56, 81, 94, 101, 108].

5.4.2 Consequences of inadequate pre-RS traceability

RE is a critical part of a software project. A lack of attention during creating and maintaining requirements may lead to a system that does not meet the stakeholder expectations [10].

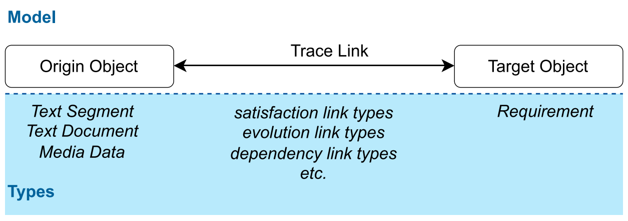

Fig. 6 Basic trace link model and types

Solving requirements issues in the early stage of a project can help avoid more expensive problems later and promote project success [9, 51, 56]. Pre-RS traceability can avoid many problems during the creation and maintenance of RS. The following list presents negative consequences of poorly or inadequately implemented pre-RS traceability:

- *"Black-box" requirements* - Requirements without any production details, such as previous versions or decisions, "[fail] to support an exploratory approach to [RE] in which requirements can emerge and evolve over time." [34]. A lack of production details becomes more critical in long-term projects, especially if responsible employees change [20, 33].
- *Expensive consequences beyond the time frame* - If requirements issues are not revealed during the creation and the maintenance before the implementation starts, it becomes more expensive to solve those problems later [37, 81]. Studies show that costs can increase five to ten times if issues have to be fixed during coding [56].

5.4.3 Solutions and suggestions

The specification of trace link models is the most frequently mentioned approach. Figure 6 presents an example of a basic trace model and summarizes different element types. Some articles present definitions of different types of trace links. Typical types of trace links to realize pre-RS traceability are *rational links* and *contribution links* [25, 39, 79, 108]. The types are used, for example, to create reports or to define rules for the creation or evaluation of links. Spanoudakis and Zisman [94] summarized all different types of trace links, but still, there is no standardization in this area.

To identify the relationship between problem codes and solution codes, we analyzed the overlap of codings and the occurrence of codings to both code types within a document. The analysis revealed four frequently addressed problems: inadequate maintenance of trace information, inadequate documentation of trace information, and two people-related problems: different interests/knowledge of different roles and too much work. All four problems can be addressed by defining trace model specifications and the usage of specific techniques for obtaining and recording trace information. We identified two different recording types: (a) the recording of all available data [36, 60, 74, 76, 110], and (b) the recording of only predefined artifacts also called selective tracing [92, 107]. Recording type (a) has to handle a large amount of data and recording type (b) requires effort to decide what has to be traceable. Researchers developed (semi-)automated approaches to reduce the workload and to avoid the loss of trace information [4, 39, 41, 42, 75, 76, 94]. The optimal, and so far unattained solution is to create correct pre-RS trace information as a by-product to reduce the workload and avoid the loss of trace and source information.

We have identified three issues that are not addressed by any solution and therefore require more research. **These are the problem of inadequate versioning support of trace links, the problem of poor adaptability for project-specific needs, and the problem of no trust in possibly incorrectly created trace links.** Adapting a trace strategy for different characteristics of the project or an organization is still a big challenge [25, 36, 75, 94].

Our literature review shows that it is necessary to develop and establish general guidelines and standards. For example, to store trace information in an exchangeable file format like "Requirements Interchange Format".[1] Such an exchange format potentially improves collaboration and information exchange inside or across company boundaries, thus increasing the awareness of organizations and users for the topic of pre-RS traceability [65, 92, 94, 108, 109].

5.5 (RQ3) What are pre-RS traceability techniques?

A lot of research has been done to invent various techniques (methods, models, etc.) including tools to support pre-RS traceability. Out of all 77 articles, 45 articles describe concrete techniques for the implementation of traceability, and 35 papers, out of these 45 papers about techniques, are relevant to answering RQ3. The other papers present solutions that build on existing pre-RS traceability techniques [4, 54, 64] or do not directly link requirements with their source [23, 28, 38, 59, 63, 92].

5.5.1 Trace techniques

Each trace technique should consider three aspects [75]:

- **Definition of trace link** specifies what artifacts to trace (incl. source artifact and target artifact). The type of relationship represents the link between these artifacts.
- **Production of trace link** describes how and when a trace link should be captured. This can be done on-line and off-line [65, 108]. Capturing a trace link **on-line** means

[1] https://www.omg.org/spec/ReqIF/1.2/

storing it automatically during traceability activities as a by-product. In the case of pre-RS traceability, a trace link to the source document will be stored while creating or maintaining a requirement. Capturing a trace link **off-line** means storing the link automatically or manually after the source and target artifact are produced. In addition, techniques distinguish between **full tracing**, tracking all artifacts, and **selective tracing**, tracing only a selection of predefined artifacts.
- **Extraction of trace link** provides one or multiple mechanisms of getting one or many desired trace links. Therefore, trace techniques should be flexible to support different use cases. An extraction mechanism can support selective tracing (filtering traces by selected patterns or characteristics), interactive tracing (browsing, guiding, and navigating through a set of trace links), and non-guided tracing (going from one artifact or a trace link to another at will).

Even if **maintenance** was not explicitly mentioned much, we would like to mention it explicitly here as part of the extraction and production of trace links. Maintaining trace links is essential to preserve traceability and thus the quality of the RS over the long term. In addition, maintenance is very important for working with trace links, because a user should be able to rely on a link being up-to-date. When a user can no longer do this, they often lose trust in all trace links and avoid using them [8, 42, 75, 94, 101, 107, 109]. No trust in trace links is one of the people-related problems presented in section 5.4.

Differences between techniques

Our investigation of the techniques revealed 16 different concepts (or dimensions) that were used to achieve pre-RS traceability and that were combined in various ways. These concepts can be divided into two groups. The first group consists of *basic concepts specifically used for traceability* (Table 9) and the other group consists of the broader *basic concepts*, that were adopted to achieve pre-RS traceability (Table 10).

An analysis of the prevalence of these concepts between 1992 and 2022 shows MDD and PBRT as the most frequently used concepts since 1995 [33]. In contrast, KyBT and ontologies have only been in use since 2003 [60, 102]. The “youngest” concepts are NLP, Blockchain and KBT [42, 56, 97, 98]. MDD is a concept that is already being used successfully in various areas of software engineering. However, lightweight concepts like KyBT or ontologies are easy to scale for growing projects, easy to understand, and do not require specific predefined rules or models. As a result, they are increasingly being used for large projects with many different involved roles [69, 81, 102].

State of evaluation

Analyzing the state of evaluation of all published techniques shows that 16 techniques are evaluated by single-case case studies, 13 tools were implemented as prototypes, and two publications present comparisons with alternative approaches. Only six techniques are evaluated by multiple-case case studies or more extensive evaluations [33, 42, 57, 74, 81]. To reduce the gap between scientific findings and industrial implementation, more extensive evaluations in an industry context are required.

5.5.2 Trace tools

We have counted 18 tools that are mentioned throughout the articles. DOORS[2] and other IBM products play a major role in realizing traceability. DOORS is known for managing requirements, including requirements traceability, and is well established in industry [20, 42, 65, 80, 100, 108]. Pro-ART, another frequently mentioned tool, focuses on pre-RS traceability [76, 80]. This tool is well known in research but has hardly been used in industry. We observe that tools established in the industry are either (a) very adaptable and support many tasks of the development process (like DOORS) or (b) are implemented specifically for a given project, including the integration into the existing tooling landscape (like MARS [100]).

6 Discussion

Previous SLRs focus on requirements traceability and report on pre-RS traceability only as a sub-topic. Other articles present solutions to solve specific aspects of the pre-RS traceability problem. To get an overview of the state-of-the-art in pre-RS traceability we conducted an SLR to synthesize prior research and draw novel insights by connecting different facets of pre-RS traceability which have been investigated so far. The code system is presented in Appendix B and a previous technical report [83]. The main goal of the technical report is to provide related articles on pre-RS traceability-related topics for researchers who want to know more about particular codes of the code system.

This SLR about pre-RS traceability is based on three research questions. The four most used main codes *use cases and benefits*, *problems and challenges*, *solutions and suggestions*, and *trace techniques* that emerged mirror the objectives of the three research questions, with problems and solutions (RQ2) being split into three main category codes. One of these categories, however, was surprising. We considered the *consequences of poor pre-RS traceability*

[2] DOORS Overview by IBM: https://www.ibm.com/docs/en/ermd/9.7.0?topic=overview-doors

Table 9 Group 1: Basic concepts specifically used to achieve traceability

Basic traceability-specific concepts	Description	No. of codings
Keyword/Annotation/Tag-based traceability (KyBT)	This technique consists of source and target artifacts or statements linked by one word, code, or tag. This technique has become particularly common in recent times, as it is often a lightweight and easily scalable approach	38
Personnel-based requirements traceability (PBRT)	These techniques take people-related behavior and individual intentions and roles into focus	31
Simple-link-base traceability (SLBT)	Simple-linking-techniques store the trace links between artifacts explicitly. It is for example done by an RT matrix	10
Goal-centric traceability (GCT) / Goal-oriented Requirements Engineering (GORE)	In the context of pre-RS traceability, it supports trace links between requirements and stakeholder goals. GCT can realize the tracing of functional and non-functional requirements	8
Value-based requirements traceability (VBRT)	Like FORT, VBRT is used mostly to trace functional requirements. This technique distinguishes between requirements that are valuable to trace and requirements with less value to trace. This type of selective tracing reduces the effort by 35% compared to full tracing [100]	6
Feature-oriented requirements traceability (FORT)	FORT is used for tracing functional requirements. It is a type of selective traceability based on feature prioritization	5
Event-based traceability (EBT)	EBT is based on the "publish-subscribe" mechanism and handles functional and non-functional requirements. For example, this technique creates trace links after a change request is executed	5
Decision-based traceability (DCT)	This technique focuses on decisions made about the architecture or functions within a software project and creates or maintains trace links	4
Rule-based approach (RB)	Reducing cost and increasing efficiency can be achieved by using rule-base traceability approaches. RB supports functional and non-functional requirements and is based on predefined rules about structures and classifications when a trace link should be created	3
Artifact-based requirements traceability (ABRT)	This technique is based on existing relationships between artifacts or different versions of artifacts	3
Knowledge-based traceability (KBT)	Based on historical changes, knowledge-based traceability uncovered relationships and potential change impacts and applies the knowledge to create and maintain future trace links	1

Table 10 Group 2: Basic concepts adopted to achieve traceability

Basic concepts	No. of codings
Model-driven development (MDD)	40
Natural language processing (NLP)	20
Ontology	12
Graph	6
Blockchain	1

important enough to warrant it being one of the main codes. Yet, in the literature we reviewed, we only found evidence in five instances of codings. The consequences of neglecting this part of requirement traceability suggest further investigation.

Pre- and post-RS traceability are closely interconnected aspects of requirements traceability, making it challenging to isolate specific pre-RS traceability-related information. However, our findings indicate that the majority of identified problems are people-related, particularly involving "Different interests/knowledge of different roles" and "Too much work." In comparison to post-RS traceability, pre-RS traceability involves a more diverse range of individuals such as customers, requirements engineers, product owners, etc., and documentation types such as recordings of interviews, meeting protocols, documentation of legacy systems, etc. The project environment and company environment emerged as two additional influential factors, as revealed by the results of RQ1. Consequently, stakeholders and researchers are interested in developing standardizations, such as defining trace link models (see results of RQ2). Further research is required to delve deeper into the influence of factors, including human factors, project environment, and company environment, in realizing pre-RS traceability. Gaining a better understanding of the impact of each factor on pre-RS traceability solutions can aid in the development of strategies that incorporate standardizations.

During our analysis, we uncovered 45 articles about techniques supporting pre-RS traceability. 25 approaches were evaluated based on a case study, field study, comparison, or implemented as prototypes. Six approaches were evaluated by multiple case studies or multiple data sets. Only five papers had elaborate evaluations. The exclusion of articles without significant evaluation would therefore have led to a considerable loss of information.

Leveraging Artificial Intelligence (AI) to enhance RE is a well-explored research domain that is gaining increased attention due to the rapid advancements in AI technologies [18]. While Kaur et al. [55] provide a comprehensive review of AI techniques for various RE tasks, they do not specifically address pre-RS traceability. The absence of AI in the presented techniques in subSect. 5.5.1 may be attributed to the current limitation in extracting precise pre-RS trace links. Inaccurate and faulty links contribute to users' lack of trust in these trace links, as discussed in Sect. 5.4.1. However, given the rapid advancement of AI technology, future research is warranted in this area to address these challenges[17].

Even though we have found plenty of benefits and use cases laid out repeatedly in the articles (Sect. 5.3) still there exists a large gap between scientific solutions and industry practice for pre-RS traceability [42, 108]. Furthermore, there are not enough evaluations in the context of industry projects. Many case studies do not correspond to the size of real projects [42, 108]. Much more research has to be done in analyzing current industry practices and how new solutions can be applied. However, it is not only research that should be following the industry, but the industry should also be open to new approaches and the corresponding additional effort required to apply them.

6.1 Future research topics

We conceive the following future research topics:

- **Influencing factors for trace strategy**: Despite the extensive research on pre-RS trace techniques, a common ground or denominator remains elusive to develop a trace strategy. Therefore, further investigation is needed to identify the influencing factors that a successful pre-RS trace strategy must consider. This includes exploring the human factor and the project environment, such as project characteristics (size, longevity, process type). Our research [83] investigates the influencing factors and adds insights based on a qualitative survey.
- **ROI of pre-RS traceability**: Kaindl et al. [50] already recommended investigating the economics of RE to narrow the gap between scientific research and industry implementation. This also applies to pre-RS traceability. As we have shown in Sect. 5.4.2, there is little research on the impact of missing pre-RS traceability or developed trace strategies that provide concrete metrics. The development and application of metrics to measure improvements and the return on investment (ROI) of specific pre-RS trace strategies would help determine the value of pre-RS traceability within projects. Understanding the ROI can also serve as a persuasive argument for industry partners to adopt and enhance pre-RS traceability strategies and techniques.
- **Standardized set of trace attributes and link types**: As demonstrated in Sect. 5.4.3, the definition of a trace link model already tackles certain issues. Further exploration of the various link types and attributes can serve as the foundation for a standardized set of trace information. A standardized set would facilitate cross-project and cross-company collaboration, promote knowledge exchange, and enhance the reusability of trace information.
- **Version support for trace attributes and links**: As projects evolve, maintaining up-to-date trace information becomes crucial but is a challenge so fare [65, 108]. Developing proper version support for trace information can assist in evaluation, preserve valuable experience, support future decision-making, and contribute to knowledge management.
- **Evaluation of trace techniques**: As presented in subSect. 5.5.1, the state of evaluation of techniques in an industry context can be improved. Further evaluation studies in an industrial project can narrow the gap between research and industrial practice.

7 Threats to validity

To identify and address threats to validity we build on Zhou et al. [111].

Construct validity

Based on much literature about requirements traceability, we found that pre-RS traceability is an important topic that is often only mentioned as a sub-topic. However, a lot of literature reported on techniques that implement pre-RS traceability. To dive deeper into this topic we created three research questions. Although we focused on the topic of pre-RS traceability, our questions cover a broad view of use cases, problems, solutions, and techniques. We want to use this to create and provide a good basis for further research. It is possible that detailed information is inadequately addressed, necessitating further research for in-depth exploration. This includes the discussion of various person-related issues or the further exploration of AI for pre-RS tractability, for instance.

Construct confounding is a possible threat to construct validity that is especially prevalent in an SLR because multiple authors may use differing terminology for the same constructs. To address this challenge we created the code "Pre-RS traceability general" consisting of subcodes to collect definitions and trace types. Together with the code "Requirements traceability", which collects information about positioning pre-RS traceability in the field of requirements traceability, we explicate our definitions of relevant constructs in Sect. 5.2 and relate them to the terms found throughout the SLR. This section should create a common understanding to address the research questions in the following sections.

Internal validity

Requirements traceability including pre-RS traceability is an area where a lot of research is going on. Since we also chose our research questions more broadly, we opted for a keyword search rather than formulating a more detailed query. Due to the possibly varying terminology used, the selection of search terms may not cover all relevant articles. We addressed this through snowballing by backward and forward searches to identify more relevant articles. In this way, we also considered articles that were not available in one of the four databases selected at the beginning. However, articles that used different terminology and was not cited within the set of selected papers may still be missing from our sample even though it may be relevant. The chance of this occurring is particularly high for newer, less frequently cited papers. Newer papers using differing terminology were still included though if they cited one of the selected papers due to the forward search in our snowballing approach. To ensure that we only included high-quality literature and by following best practices, we only included peer-reviewed literature. This means that relevant blog articles about pre-RS trace techniques and experiences, for example from industry, are not taken into account.

As mentioned in the previous Sect. 6, a challenge is to distinguish papers about pre-RS traceability and requirements traceability in general. This was particularly difficult in the selection of papers on certain techniques to realize pre-RS traceability. Therefore, we selected only papers about techniques to trace between requirements specification and source artifacts. Where the source artifact is represented by different instances of stakeholder information such as the name of the stakeholder, a note about a reason, or a document about a standard.

External validity

We included articles published between 1992 and 2022. Articles published outside of this period may affect the generalizability of the SLR results. Most of the papers come from the field of software development because the topic is particularly present there. Nevertheless, the topic of pre-RS traceability is also relevant for the development of hardware. Especially with safety-critical hardware, where standards must be taken into account. In our research, we focused on the creation of RS and not on the creation of user stories which are typically forms to document requirements in agile projects. Realizing pre-RS traceability when creating and maintaining user stories entails further challenges in terms of the speed of information [12, 95]. Unfortunately, we cannot make any statements about pre-RS traceability in the context of hardware-related projects or creating user stories, but only provide transferable information. Further research is needed at this point.

Conclusion validity

The degree to which the conclusions about the relationships between our results and the research questions are valid relies on the adherence to established research methods and a good fit of the research design to the research questions. To that end, the whole research process was defined and documented by a research protocol, serving as an audit trail. The research protocol was continuously reviewed by an experienced researcher within peer debriefing sessions.

8 Conclusion

Pre-requirement specification (pre-RS) traceability is the ability to link requirements to their source. Knowledge about requirements creation and who was involved in it can have a significant impact on project success. Therefore, we conducted an SLR about pre-RS traceability to provide an overview of the state-of-the-art. Compared to post-RS traceability, we found significantly less research was done specifically on pre-RS traceability. Pre-RS traceability is often mentioned as a sub-topic in previous SLRs on requirements traceability.

To structure our research we developed three research questions to capture information about use cases, benefits, problems, and current solutions. We included the consequences of inadequate pre-RS traceability mentioned in the articles to enrich this topic because it is often seen as a requirements trace activity with high effort and comparatively few benefits.

We used qualitative data analysis (QDA) to process 77 relevant papers. The resulting code system, including all references, is presented in the previously published technical report [83] and Appendix B.

Since we have identified slightly different definitions of terms, the article presents the definitions that emerge from the common consensus of previous research.

The development of a suitable pre-RS trace strategy is a significant challenge. Three influencing factors have to be considered the environment, the users, and the usage goal of trace links. The most frequently mentioned problems are related to the trace user, such as accommodating different interests of different roles or doing too much work for unseen benefits. More research is needed to address these problems. Much more research has been done in developing techniques to support pre-RS traceability. But still, there exists a gap between scientific solutions and industry implementations. This work exposes this gap and recommends further research directions to narrow this gap.

Appendix A List of included articles

The list of included articles is sorted alphabetically. The table of articles, descried in Sect. 4.4, is present as document [73].

Leite and Oliveira [61] A client-oriented requirements baseline
Botaschanjan et al. [7] A conceptual model for requirements engineering and management for change-intensive software
España et al. [27] A domain specific language for data-centric infographics
Glinz [31] A glossary of requirements engineering terminology
Dubois et al. [22] A Model for Requirements Traceability in a Heterogeneous Model-Based Design Process: Application to Automotive Embedded Systems
El Ghazi and Assar[25] A multi-view-based traceability management method
Wood et al. [110] A multimedia approach to requirements capture and modeling
Nair et al. [66] A review of traceability research at the requirements engineering conferencere@21
Serrano anf do Prado Leite [88] A rich traceability model for social interactions
Winkler and Pilgrim [108] A survey of traceability in requirements engineering and model-driven development
Bouillon et al. [8] A Survey on Usage Scenarios for Requirements Traceability in Practice
Tufail et al. [101] A systematic review of requirement traceability techniques and tools
Jayatilleke and Lai [47] A systematic review of requirements change management
Hayes et al. [42] Advancing candidate link generation for requirements tracing: the study of methods
Urrego-Giraldo [102] Agent-based knowledge keep tracking
Lee et al. [60] An Agile Approach to Capturing Requirements and Traceability
Gotel and Finkelstein [37] An analysis of the requirements traceability problem
Rempel et al. [82] An empirical study on project-specific traceability strategies
Pinheiro and Goguen [74] An object-oriented tool for tracing requirements
Souali et al. [91] An overview of traceability: Definitions and techniques
Souali et al. [92] An Overview of Traceability: Toward a general multi-domain model
Espinoza et al. [28] Analyzing and Systematizing Current Traceability Schemas
Ossher et al. [70] Business insight toolkit: Flexible pre-requirements modeling
Wohlrab et al. [109] Collaborative Traceability Management: Challenges and Opportunities
Gotel and Finkelstein [33] Contribution structures [Requirements artifacts]
Cleland-Huang et al. [15] Decision-Centric Traceability of architectural concerns
Ahmad and Ghazali [1] Documenting Requirements Traceability Information for Small Projects
Stone and Sawyer [97] Exposing Tacit Knowledge via Pre-Requirements Tracing
Gotel and Finkelstein [35] Extended requirements traceability: results of an industrial case study
Ramesh [78] Factors influencing requirements traceability practice
Weber-Jahnke and Onabajo [104] Finding Defects in Natural Language Confidentiality Requirements
Ossher et al. [71] Flexible modeling tools for pre-requirements analysis: conceptual architecture and research challenges
Kitapci and Boehm[57] Formalizing Informal Stakeholder Decisions–A Hybrid Method Approach
Liang et al. [62] From collective knowledge to intelligence: pre-requirements analysis of large and complex systems
Rocky et al. [85] Hierarchical Permissioned Blockchain and Traceability Of Requirement Changes
Aleixo et al. [3] Identification and classification of barriers and benefits of requirements traceability in project development
Stone and Sawyer [98] Identifying tacit knowledge-based requirements
Imtiaz et al. [46] Impact Analysis from Multiple Perspectives: Evaluation of Traceability Techniques
Hayes et al. [41] Improving requirements tracing via information retrieval
Kaufmann and Riehle [53] Improving Traceability of Requirements Through Qualitative Data Analysis
Shukla et al. [89] Integrated requirement traceability, multiview modeling, and decision-making: A systems engineering approach for integrating processes and product
Gacitua et al. [77] Making Tacit Requirements Explicit
Mohan and Ramesh [64] Managing variability with traceability in product and service families
Mäder et al. [65] Motivation Matters in the Traceability Trenches
Assawamekin et al. [5] Ontology-based multiperspective requirements traceability framework
Ravichandar et al. [81] Pre-Requirement Specification Traceability: Bridging the Complexity Gap through Capabilities

Pohl [76] PRO-ART: enabling requirements pre-traceability
Sawyer et al. [86] Profiling and Tracing Stakeholder Need
He and Li [43] RE_PROV: Modeling Requirement Provenance with PROV
Al-walidi et al. [2] Recommender Systems in Requirements Engineering: A Systematic Literature Review
Chikh and Aldayel [11] Reengineering Requirements Specification Based on IEEE 830 Standard and Traceability
Grunbacher et al. [38] Repeatable quality assurance techniques for requirements negotiations
Dick et al. [20] Requirements Engineering (Fourth Edition)
Finkelstein [30] Requirements engineering: a review and research agenda
Gotel [36] Requirements Traceability
Pinheiro [75] Requirements Traceability
Castro et al. [10] Requirements Traceability in Agent-Oriented Development
Wibowo and Davis [106] Requirements Traceability ontology to support requirements management
Torkar et al. [100] Requirements traceability state-of-the-art: a systematic review and industry case study
Ramesh et al. [80] Requirements traceability: Theory and practice
Gotel and Finkelstein [34] Revisiting requirements production
Wiegers and Beatty [107] Software requirements
Spanoudakis and Zisman [94] Software traceability: a roadmap
Duggal et al. [23] SRS Automator - An Attempt to Simplify Software Development Lifecycle
Panis [72] Successful Deployment of Requirements Traceability in a Commercial Engineering Organization... Really
Kaufmann and Riehle [54] The QDAcity-RE method for structural domain modeling using qualitative data analysis
Ramesh and Jarke [79] Toward reference models for requirements traceability
Haidrar et al. [39] Toward a generic framework for requirements traceability management for SysML language
Laurent et al. [59] Toward Automated Requirements Triage
Nair et al. [67] Traceability Research at the Requirements Engineering Conference: Results and Extracted Data
Bashir and Qadir [6] Traceability techniques: A critical study
Hao and Jaafar [40] Tracing user interface design pre-requirement to generate interface design specification
Kitapci and Boehm [56] Using a Hybrid Method for Formalizing Informal Stakeholder Requirements Inputs
Ossher et al. [69] Using tagging to identify and organize concerns during pre-requirements analysis
Altaf et al. [4] Visualization representing benefits of pre-requirement specification traceability
Lohmann et al. [63]Web Platform for Social Requirements Engineering
Kannenberg and Saiedian [52] Why software requirements traceability remains a challenge.

Appendix B Code System

The codes of the first level of the code system are arranged according to topic, and the codes of the sub-level are arranged according to the descending number of the codings. The code system including further descriptions and references is presented in our technical report [83].

L1 **Requirements Traceability** *(Codings: 5)*
L1 **Pre-RS traceability general** *(Group code)*

- L2 Definition/Descriptions *(Codings: 20)*
  - L3 Including inter-requirements traceability *(Codings: 4)*
  - L3 Exclude inter-requirements traceability *(Codings: 2)*
- L2 Delimitation of Types *(Codings: 15)*
  - L3 Inter-requirements traceability *(Codings: 6)*
  - L3 Pre-RS forwards traceability *(Codings: 5)*
  - L3 Pre-RS backwards traceability *(Codings: 5)*
  - L3 Requirement-Stakeholder/Roles *(Codings: 3)*
  - L3 Requirement-Rationale *(Codings: 3)*
  - L3 Vertical traceability *(Codings: 2)*
  - L3 Upstream *(Codings: 2)*
  - L3 Multiplicity between origin and target *(Codings: 2)*
  - L3 Horizontal traceability *(Codings: 2)*
  - L3 Non-functional tracing *(Codings: 1)*
  - L3 Functional tracing *(Codings: 1)*
  - L3 Extra-requirements traceability *(Codings: 1)*
- L2 Traceability activities *(Codings: 7)*

L1 **Users applying pre-RS traceability** *(Codings: 4)*

- L2 User types by motivation/practice *(Codings: 3)*
  - L3 High-end user *(Codings: 10)*
  - L3 Low-end user *(Codings: 8)*

L2 End-user *(Codings: 3)*
L2 Provider *(Codings: 3)*

L1 **Use cases and benefits** *(Group code)*

L2 Use cases *(Group code)*

L3 Finding Source (support understanding) *(Codings: 40)*
L3 Responsibility identification *(Codings: 18)*
L3 Fulfilment of regulatory compliance/norms *(Codings: 18)*
L3 Prove fulfilment of stakeholder needs *(Codings: 16)*
L3 Getting state of RS *(Codings: 11)*
L3 Manage system evolution (maintaining) *(Codings: 11)*
L3 Keeping track of history/relationships *(Codings: 11)*
L3 Change-management *(Codings: 9)*

L4 Impact Analysis *(Codings: 17)*
L4 Capture and support decisions *(Codings: 15)*

L3 Knowledge management system *(Codings: 7)*
L3 Requirements prioritization *(Codings: 5)*
L3 Requirements negotiations *(Codings: 5)*

L2 Benefits *(Group code)*

L3 Monitor and gain knowledge for future *(Codings: 21)*
L3 Improve Product/SW quality *(Codings: 13)*
L3 Support reusability of requirements *(Codings: 10)*
L3 Improve communication and collaboration *(Codings: 8)*
L3 Satisfaction of stakeholders *(Codings: 4)*
L3 Reveal tacit knowledge *(Codings: 4)*
L3 Finding missing requirements *(Codings: 2)*
L3 Reduction of maintenance costs *(Codings: 2)*
L3 Finding unnecessary requirements *(Codings: 2)*

L1 **Problems and challenges** *(Codings: 5)*

L2 Person-related problems *(Codings: 4)*

L3 Different interests/knowledge of different roles *(Codings: 24)*

L4 Fear of disclosure *(Codings: 3)*

L3 Too much work (vs benefits) *(Codings: 22)*
L3 No trust in traces *(Codings: 7)*
L3 Responsibility problem *(Codings: 7)*
L3 No immediate benefit seen *(Codings: 4)*
L3 Subjective or idealized traces *(Codings: 2)*

L2 Inadequate documentation of trace information *(Codings: 6)*

L3 Tacit knowledge or unnoticed links *(Codings: 11)*
L3 Ad-hoc effort *(Codings: 9)*
L3 No documentation of verbal communication or interaction *(Codings: 5)*
L3 Lack verifying correctness, completeness *(Codings: 3)*

L2 No/poor maintenance of trace information (evaluation) *(Codings: 16)*

L3 Path ephemerality problem *(Codings: 11)*

L2 No generalization/standards *(Codings: 24)*
L2 Organizational problems *(Codings: 8)*

L3 Collaboration across boundaries *(Codings: 9)*
L3 Low priority of traceability *(Codings: 7)*

L2 Tooling problems *(Codings: 8)*

L3 Exchange between tools *(Codings: 3)*
L3 Limited support for interaction or collaboration *(Codings: 2)*

L2 No/poor identification of trace artifacts *(Codings: 9)*

L3 No/poor connecting different object types *(Codings: 3)*
L3 Heterogneity of terms *(Codings: 1)*

L2 No/poor (semi-)automation *(Codings: 12)*
L2 Bad access/presentation of trace information *(Codings: 11)*
L2 Organize unstructured information *(Codings: 10)*
L2 Missing usage goal for traces *(Codings: 5)*

L3 Trace Path Suitability Problem *(Codings: 3)*

L2 No/inadequate support managing large/distributed amount of data *(Codings: 5)*
L2 Poor adaptability for project-specific needs *(Codings: 5)*
L2 No/poor versioning support of traces *(Codings: 4)*
L2 No/poor reusability *(Codings: 2)*

L1 **Consequences of poor pre-RS traceability** *(Codings: 5)*

L1 **Solutions and suggestions** *(Codings: 1)*

- L2 Solutions regarding trace link specifications *(Codings: 8)*
  - L3 Distinction of different link types *(Codings: 46)*
    - L4 Support of non-functional requirements trace types *(Codings: 2)*
  - L3 Support different object types *(Codings: 8)*
  - L3 Define trace content *(Codings: 4)*
- L2 Person-related solutions *(Group code)*
  - L3 Support collaborative work and communication *(Codings: 8)*
    - L4 Support different involved (social) roles *(Codings: 11)*
    - L4 Provide contact information *(Codings: 2)*
  - L3 Increase awareness of traceability need *(Codings: 11)*
    - L4 Establish new trace responsibility role *(Codings: 5)*
    - L4 Provide immediate benefits to increase motivation *(Codings: 2)*
    - L4 Provide immediate benefits to increase motivation *(Codings: 2)*
  - L3 Guide user to create trace link *(Codings: 7)*
  - L3 Support of flexible use *(Codings: 1)*
  - L3 Need Understanding by Stakeholder *(Codings: 1)*
- L2 Obtain and record trace information *(Codings: 5)*
  - L3 Support (semi-)automatic Trace-Link capturing *(Codings: 9)*
  - L3 Predefined trace information *(Codings: 6)*
  - L3 Support large trace sets *(Codings: 5)*
  - L3 Description-fields on requirement *(Codings: 2)*
  - L3 Recording types *(Group code)*
    - L4 Record all available information *(Codings: 8)*
    - L4 Priority traceability *(Codings: 2)*
- L2 Enable valuable visualizations and presentations *(Codings: 18)*
- L2 Provide flexible tools/support *(Codings: 17)*
- L2 Create/improve the structure of RS *(Codings: 1)*
  - L3 Hierarchical structuring of requirements *(Codings: 6)*
  - L3 Classification/categorization of Requirements *(Codings: 3)*
  - L3 Highlight of key requirements *(Codings: 3)*
  - L3 Highlight of key requirements *(Codings: 3)*
- L2 Improve generic trace model *(Codings: 4)*
  - L3 Define a Traceability Schema *(Codings: 8)*
  - L3 Define trace preconditions *(Codings: 3)*
- L2 Provide fitting storage for traces and data *(Codings: 4)*
  - L3 Establish central storage or repository *(Codings: 9)*
- L2 Organize individual (origin) artifacts *(Codings: 10)*
- L2 Establish templates and guidelines *(Codings: 8)*
- L2 Maintain trace links and artifacts *(Codings: 2)*
  - L3 Design a modular viable (trace) system *(Codings: 3)*
  - L3 Establish periodically audits/quality checks *(Codings: 1)*
  - L3 Notify responsible person in case of change *(Codings: 1)*
- L2 Configurable traceability strategies *(Codings: 3)*
- L2 Analyze and consider special attributes *(Codings: 2)*
- L2 Support information transfer *(Codings: 1)*

L1 **Trace techniques** *(Codings: 2)*

- L2 Distinction based on traceability types *(Group code)*
  - L3 Personnel-based RT *(Codings: 5)*
    - L4 Contribution structure *(Codings: 14)*
    - L4 Agent-Based Knowledge *(Codings: 9)*
    - L4 Usage Centered Technique approach *(Codings: 3)*
  - L3 Simple traceability links *(Codings: 2)*
    - L4 RT Matrix *(Codings: 7)*
    - L4 Hyper-linked documents *(Codings: 1)*
  - L3 Goal-centric traceability (GCT) *(Codings: 8)*
  - L3 Value-based RT (VBRT) *(Codings: 6)*

L3 Feature-oriented RT (FORT) *(Codings: 5)*
L3 Event-based RT (EBT) *(Codings: 5)*
L3 Decision-based Traceability (DCT) *(Codings: 4)*
L3 Rule-based approach (RB) *(Codings: 3)*
L3 Artifact-based RT *(Codings: 3)*
L3 Knowledge-based techniques *(Codings: 1)*

L2 Model-Driven Development *(Codings: 2)*

L3 Meta Models *(Codings: 24)*
L3 Concept Model *(Codings: 8)*
L3 Hypertext model *(Codings: 5)*
L3 Richer traceability Model *(Codings: 1)*

L2 Connection by codes/words/tags *(Group code)*

L3 Qualitative Data Analysis *(Codings: 19)*
L3 Tagging *(Codings: 10)*
L3 Capabilities-based development *(Codings: 7)*
L3 Blockchain *(Codings: 3)*

L2 Natural language processing (NLP) *(Codings: 16)*

L3 Latent Semantic Analysis (LSA) *(Codings: 3)*
L3 Shallow NLP *(Codings: 1)*

L2 Trace model aspects *(Codings: 1)*

L3 Link production *(Codings: 5)*

L4 on-line *(Codings: 4)*
L4 off-line *(Codings: 1)*

L3 Definition of trace *(Codings: 3)*
L3 Link extraction *(Codings: 2)*

L2 Ontology *(Codings: 12)*
L2 Graphs *(Codings: 6)*

L1 **Traceability tool** *(Codings: 5)*

L2 From IBM *(Group code)*

L3 Doors *(Codings: 8)*
L3 Rational RequisitePro *(Codings: 4)*
L3 MARS *(Codings: 1)*

L2 Requirements tracing on-target (RETRO) *(Codings: 5)*
L2 Traceability of Object-Oriented Requirements (TOOR) *(Codings: 5)*
L2 Business Insight Toolkit (BITKit) *(Codings: 3)*
L2 Advanced Multimedia Organizer for Requirements Elicitation *(Codings: 2)*
L2 Echo *(Codings: 2)*
L2 Pro-ART *(Codings: 2)*
L2 Information Engineering Facility (IEF) *(Codings: 2)*
L2 Requirements Traceability Manager (RTM) *(Codings: 2)*
L2 Hyperledger Fabric *(Codings: 1)*
L2 Integrated System Design TOOL (ISDT) *(Codings: 1)*
L2 SuperTracePlus (STP) *(Codings: 1)*
L2 TRAM *(Codings: 1)*
L2 DesignTrack *(Codings: 1)*
L2 Teamcenter Systems Engineering tool (TcSE) *(Codings: 1)*
L2 Tool combinations *(Codings: 1)*

Acknowledgements This work was supported by a grant from Software Campus through the Federal Ministry of Education and Research[3] We would like to thank Nikolay Harutyunyan and the team of QDAcity[4] for the valuable feedback on this article and Rita Rokeya for proof reading.

Funding Open Access funding enabled and organized by Projekt DEAL. This work was supported by a grant from Software Campus through the Federal Ministry of Education and Research.

Data availability The whole code system of the qualitative data analysis is presented in the appendix and in our technical report [83]. The technical report includes all references to the literature. The table of literature, we described in the method section is available as PDF [73]. If the PDF is no longer available via this link, please contact the first author.

Code availability Code of all tools used in this study are proprietary. We used MAXQDA for the qualitative data analysis.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

[3] BMBF: Bundesministerium fuer Bildung und Forschung https://www.bmbf.de/en/index.html.

[4] QDAcity: https://qdacity.com/

References

1. Ahmad A, Ghazali MA (2007) Documenting Requirements Traceability Information for Small Projects. In: 2007 IEEE international multitopic conference, IEEE, Lahore, pp 1–5
2. Al-walidi NH, Mahmood MA, Ramadan N (2019) Recommender systems in requirements engineering: a systematic literature review. In: Proceedings of the 54th annual conference on statistics, computer sciences and operation research
3. Aleixo SR, Campese C, Mascarenhas J (2021) Identification and classification of barriers and benefits of requirements traceability in project development. World J Adv Eng Technol Sci
4. Altaf S, Shah A, Imtiaz N, Shah AS, Ahmed SF (2018) Visualization representing benefits of pre-requirement specification traceability. Int J Eng Technol 7:44
5. Assawamekin N, Sunetnanta T, Pluempitiwiriyawej C (2010) Ontology-based multiperspective requirements traceability framework. Knowl Inf Syst 25(3):493–522
6. Bashir MF, Qadir MA (2006) Traceability techniques: a critical study. In: 2006 IEEE international multitopic conference, IEEE, pp 265–268
7. Botaschanjan J, Fleischmann A, Pister M (2004) A conceptual model for requirements engineering and management for change-intensive software. In: IASTED conf. on software engineering, pp 36–41
8. Bouillon E, Mäder P, Philippow I (2013) A survey on usage scenarios for requirements traceability in practice, lecture notes in computer science. In: Doerr J, Opdahl AL (eds) Requirements engineering: foundation for software quality. Springer, Berlin, Heidelberg, pp 158–173
9. Carson RS (1998) Requirements completeness: a deterministic approach. INCOSE Int Symp 8(1):738–746
10. Castro J, Pinto R, Castor A, Mylopoulos J (2002) Requirements traceability in agent oriented development, lecture notes in computer science. In: Garcia A, Lucena C, Zambonelli F, Omicini A, Castro J (eds) Software engineering for large-scale multi-agent systems. Springer, Berlin, Heidelberg, pp 57–72
11. Chikh A, Aldayel M (2014) Reengineering requirements specification based on IEEE 830 standard and traceability. In: Rocha À, Correia AM, Tan FB, Stroetmann KA (eds) New perspectives in information systems and technologies, volume 1, advances in intelligent systems and computing. Springer, Cham, pp 211–227
12. Cleland-Huang J (2011) Traceability in agile projects. In: Cleland-Huang J, Gotel O, Zisman A (eds) Software and systems traceability. Springer, Berlin, pp 265–275
13. Cleland-Huang J, Berenbach B, Clark S, Settimi R, Romanova E (2007) Best practices for automated traceability. Computer 40(6):27–35
14. Cleland-Huang J, Gotel OC, Huffman Hayes J, Mäder P, Zisman A (2014) Software traceability: trends and future directions. In: Future of software engineering proceedings, ACM, pp 55–69
15. Cleland-Huang J, Mirakhorli M, Czauderna A, Wieloch M (2013) Decision-centric traceability of architectural concerns. In: 2013 7th international workshop on traceability in emerging forms of software engineering (TEFSE), pp 5–11. ISSN: 2157-2194
16. Corbin JM, Strauss A (1990) Grounded theory research: procedures, canons, and evaluative criteria. Qual Soc 13(1):3–21
17. Dalpiaz F (2022) Keynote-requirements conversations: a new frontier in ai-for-re. In: 2022 IEEE 30th international requirements engineering conference workshops (REW), IEEE, pp 142–142
18. Dalpiaz F, Niu N (2020) Requirements engineering in the days of artificial intelligence. IEEE Softw 37(4):7–10
19. Debnath S, Spoletini P, Ferrari A (2021) From ideas to expressed needs: an empirical study on the evolution of requirements during elicitation. In: 2021 IEEE 29th international requirements engineering conference (RE), IEEE, pp 233–244
20. Dick J, Hull E, Jackson K (2017) Requirements Engineering, 4th edn. Springer, Switzerland
21. Duarte AMD, Duarte D, Thiry M (2016) Tracebok: toward a software requirements traceability body of knowledge. In: 2016 IEEE 24th international requirements engineering conference (RE), IEEE, pp 236–245
22. Dubois H, Peraldi-Frati MA, Lakhal F (2010) A model for requirements traceability in a heterogeneous model-based design process: application to automotive embedded systems. In: 2010 15th IEEE international conference on engineering of complex computer systems, IEEE, Oxford, pp 233–242
23. Duggal M, Saxena N, Gurve M (2020) srs automator—an attempt to simplify software development lifecycle. In: 2020 6th international conference on signal processing and communication (ICSC), pp 278–283
24. Ebner G, Kaindl H (2002) Tracing all around in reengineering. IEEE Softw 19(3):70–77
25. El Ghazi H, Assar S (2008) A multi view based traceability management method. In: 2008 second international conference on research challenges in information science, IEEE, Marrakech, Morocco, , pp 393–400
26. Escalona MJ, Koch N, Garcia-Borgoñon L (2022) Lean requirements traceability automation enabled by model-driven engineering. PeerJ Comput Sci 8:e817
27. España S, Ramautar V, Overbeek S, Derikx T (2022) A domain specific language for data-centric infographics: technical report. Tech. Rep. arXiv:2203.09292
28. Espinoza A, Alarcon PP, Garbajosa J (2006) Analyzing and systematizing current traceability schemas. In: 2006 30th annual IEEE/NASA software engineering workshop, pp 21–32
29. Fernández DM, Wagner S, Kalinowski M, Felderer M, Mafra P, Vetrò A, Conte T, Christiansson MT, Greer D, Lassenius C, Männistö T, Nayabi M, Oivo M, Penzenstadler B, Pfahl D, Prikladnicki R, Ruhe G, Schekelmann A, Sen S, Spinola R, Tuzcu A, de la Vara JL, Wieringa R (2017) Naming the pain in requirements engineering. Empir Softw Eng 22(5):2298–2338
30. Finkelstein A (1994) Requirements engineering: a review and research agenda. In: Proceedings of 1st Asia-pacific software engineering conference, IEEE, Tokyo, pp 10–19
31. Glinz M (2020) Standard glossary for the certified professional for requirements engineering (cpre) studies and exam v2. Int. Requirements Engineering Board eV, Tech. Rep
32. Goguen J (1993) Social issues in requirements engineering. In: [1993] Proceedings of the IEEE international symposium on requirements engineering, IEEE, San Diego, CA, pp 194–195
33. Gotel O, Finkelstein A (1995) Contribution structures [Requirements artifacts]. In: Proceedings of 1995 IEEE international symposium on requirements engineering (RE'95), IEEE, York, pp100–107
34. Gotel O, Finkelstein A (1996) Revisiting requirements production. Softw Eng J 11(3):166–182
35. Gotel O, Finkelstein A (1997) Extended requirements traceability: results of an industrial case study. In: Proceedings of ISRE '97: 3rd IEEE international symposium on requirements engineering, IEEE, Annapolis, MD, pp 169–178
36. Gotel OCZ (1992) Requirements Traceability. Main Report, Centre for Requirements and Foundations, Oxford University Computing Lab, Oxford, Oxford
37. Gotel OCZ, Finkelstein CW (1994) An analysis of the requirements traceability problem. In: Proceedings of IEEE international conference on requirements engineering, pp 94–101
38. Grunbacher P, Halling M, Biffl S, Kitapci H, Boehm B (2003) Repeatable quality assurance techniques for requirements negotiations. In: 36th Annual Hawaii international conference on

system sciences, 2003. Proceedings of the, IEEE, Big Island, HI, p 9

39. Haidrar S, Anwar A, Roudies O (2016) Towards a generic framework for requirements traceability management for SysML language. In: 2016 4th IEEE international colloquium on information science and technology (CiSt), IEEE, Tangier, Morocco, pp 210–215
40. Hao HM, Jaafar A (2009) Tracing user interface design pre-requirement to generate interface design specification. In: 2009 international conference on electrical engineering and informatics, vol 01, IEEE, Selangor, pp 287–292
41. Hayes J, Dekhtyar A, Osborne J (2003) Improving requirements tracing via information retrieval. Proceedings. 11th IEEE international requirements engineering conference, 2003. IEEE, Monterey Bay, CA, pp 138–147
42. Hayes J, Dekhtyar A, Sundaram S (2006) Advancing candidate link generation for requirements tracing: the study of methods. IEEE Trans Softw Eng 32(1):4–19
43. He Y, Li X (2016) RE_prov: Modeling requirement provenance with PROV. In: 2016 23rd Asia-Pacific software engineering conference (APSEC), IEEE, Hamilton, pp 397–400
44. Heck P, Zaidman A (2018) A systematic literature review on quality criteria for agile requirements specifications. Softw Qual J 26:127–160
45. Hübner P, Paech B (2020) Interaction-based creation and maintenance of continuously usable trace links between requirements and source code. Empir Softw Eng 25(5):4350–4377
46. Imtiaz S, Ikram N, Imtiaz S (2008) Impact analysis from multiple perspectives: evaluation of traceability techniques. In: 2008 The third international conference on software engineering advances, IEEE, Sliema, Malta, pp 457–464
47. Jayatilleke S, Lai R (2018) A systematic review of requirements change management. Inf Softw Technol 93:163–185
48. Kaindl H (1993) The missing link in requirements engineering. ACM SIGSOFT Softw Eng Notes 18(2):30–39
49. Kaindl H (2001) Using hypermedia in requirements engineering practice. New review of hypermedia and multimedia 7(1):185–205
50. Kaindl H, Brinkkemper S, Bubenko JA Jr, Farbey B, Greenspan SJ, Heitmeyer CL, Leite JCSDP, Mead NR, Mylopoulos J, Siddiqi J (2002) Requirements engineering and technology transfer: obstacles, incentives and improvement agenda. Requir Eng 7:113–123
51. Kamalrudin M, Hosking J, Grundy J (2011) Improving requirements quality using essential use case interaction patterns. In: 2011 33rd international conference on software engineering (ICSE), IEEE, pp 531–540
52. Kannenberg A, Saiedian H (2009) Why software requirements traceability remains a challenge. J Defense Softw Eng 22:14–19
53. Kaufmann A, Riehle D (2015) Improving traceability of requirements through qualitative data analysis. In: Proceedings of the software engineering 2015
54. Kaufmann A, Riehle D (2019) The QDAcity-RE method for structural domain modeling using qualitative data analysis. Requir Eng 24(1):85–102
55. Kaur K, Singh P, Kaur P (2020) A review of artificial intelligence techniques for requirement engineering. In: Computational methods and data engineering: Proceedings of ICMDE 2020, vol 2 pp 259–278
56. Kitapci H, Boehm BW (2006) Using a Hybrid Method for Formalizing Informal Stakeholder Requirements Inputs. In: Fourth international workshop on comparative evaluation in requirements engineering (CERE'06 - RE'06 Workshop), IEEE, pp 48–59
57. Kitapci H, Boehm BW (2007) Formalizing Informal Stakeholder Decisions-A Hybrid Method Approach. In: 2007 40th annual hawaii international conference on system sciences (HICSS'07), IEEE, Waikoloa, HI, pp 283c–283c
58. Kitchenham B (2004) Procedures for performing systematic reviews. Technical Report 1.0, Keele, UK, Keele University
59. Laurent P, Cleland-Huang J, Duan C (2007) Towards automated requirements triage. In: 15th IEEE international requirements engineering conference (RE 2007), IEEE, Delhi, pp 131–140
60. Lee C, Guadagno L, Jia X (2003) An agile approach to capturing requirements and traceability. In: Proceedings of the 2nd international workshop on traceability in emerging forms of software engineering (TEFSE 2003), vol 20
61. Leite J, Oliveira A (1995) A client oriented requirements baseline. In: Proceedings of 1995 IEEE international symposium on requirements engineering (RE'95), IEEE, York, pp 108–115
62. Liang P, Avgeriou P, He K, Xu L (2010) From collective knowledge to intelligence: pre-requirements analysis of large and complex systems. In: Proceedings of the 1st Workshop on Web 2.0 for software engineering (Web2SE), ACM, pp 26–30
63. Lohmann S, Dietzold S, Heim P, Heino N (2009) A web platform for social requirements engineering. In: Software engineering 2009—workshopband. Gesellschaft für Informatik e.V. . Accepted: 2019-02-20T10:13:00Z ISSN: 1617-5468
64. Mohan K, Ramesh B (2002) Managing variability with traceability in product and service families. In: Proceedings of the 35th annual hawaii international conference on system sciences, IEEE, Big Island, HI, pp 1309–1317
65. Mäder P, Gotel O, Philippow I (2009) Motivation matters in the traceability trenches. In: 2009 17th IEEE international requirements engineering conference, pp 143–148
66. Nair S, Vara JLdl, Sen S (2013) A review of traceability research at the requirements engineering conferencere@21. In: 2013 21st IEEE international requirements engineering conference (RE), IEEE, Rio de Janeiro, pp 222–229
67. Nair S, Vara JLDL, Sen S, Surface SN (2013) Traceability research at the requirements engineering conference: results and extracted data. Technical Report 2012-22, Simula Research Laboratory
68. Nicolás J, Toval A (2009) On the generation of requirements specifications from software engineering models: a systematic literature review. Inf Softw Technol 51(9):1291–1307
69. Ossher H, Amid D, Anaby-Tavor A, Bellamy R, Callery M, Desmond M, De Vries J, Fisher A, Krasikov S, Simmonds I, Swart C (2009) Using tagging to identify and organize concerns during pre-requirements analysis. In: 2009 ICSE workshop on aspect-oriented requirements engineering and architecture design, IEEE, Vancouver, BC, pp 25–30
70. Ossher H, Bellamy R, Amid D, Anaby-Tavor A, Callery M, Desmond M, de Vries J, Fisher A, Frauenhofer T, Krasikov S, Simmonds I, Swart C (2009) Business insight toolkit: flexible pre-requirements modeling. In: 2009 31st international conference on software engineering—Companion Volume, IEEE, Vancouver, BC, pp 423–424
71. Ossher H, Bellamy R, Simmonds I, Amid D, Anaby-Tavor A, Callery M, Desmond M, de Vries J, Fisher A, Krasikov S (2010) Flexible modeling tools for pre-requirements analysis: conceptual architecture and research challenges. In: Proceedings of the ACM international conference on Object oriented programming systems languages and applications, OOPSLA '10, Association for Computing Machinery, Reno/Tahoe, Nevada, pp 848–864
72. Panis MC (2010) Successful Deployment of Requirements Traceability in a Commercial Engineering Organization...Really. In: Proceedings of the 2010 18th IEEE international requirements engineering conference, RE '10, IEEE Computer Society, Washington, DC, pp 303–307

73. Mucha J (2023) Table of articles. https://github.com/Juliiia/my_page/blob/main/src/assets/documents/2023_Mucha_TableOfArticles_SLR_pre-RSTraceability.pdf
74. Pinheiro F, Goguen J (1996) An object-oriented tool for tracing requirements. IEEE Softw 13(2):52–64
75. Pinheiro FAC (2004) Requirements traceability. in: perspectives on software requirements, Springer, pp 91–113
76. Pohl K (1996) PRO-ART: enabling requirements pre-traceability. In: Proceedings of the second international conference on requirements engineering, pp 76–84
77. Gacitua R, Ma L, Nuseibeh B, Piwek P, De Roeck AN, Rouncefield M, Sawyer P, Willis A, Yang H, (2009) Making Tacit Requirements Explicit. In: 2009 second international workshop on managing requirements knowledge, IEEE, Atlanta, GA, pp 40–44
78. Ramesh B (1998) Factors influencing requirements traceability practice. Commun ACM 41(12):37–44
79. Ramesh B, Jarke M (2001) Toward reference models for requirements traceability. IEEE Trans Softw Eng 27(1):58–93
80. Ramesh B, Stubbs C, Powers T, Edwards M (1997) Requirements traceability: theory and practice. Ann Softw Eng 3(1–2–3–4):397–415
81. Ravichandar R, Arthur JD, Pérez-Quiñones M (2007) Pre-requirement specification traceability: bridging the complexity gap through capabilities. International symposium on grand challenges in traceability, TEFSE/GCT 2007:10
82. Rempel P, Mäder P, Kuschke T (2013) An empirical study on project-specific traceability strategies. In: 2013 21st IEEE international requirements engineering conference (RE), IEEE, Rio de Janeiro, pp 195–204
83. Krause J, Kaufmann A, Riehle D (2020) The code system of a systematic literature review on pre-requirements specification traceability. Tech. Rep.
84. Krause J, Kaufmann A, Riehle D (2022) The benefits of pre-requirements specification traceability. In: 2022 30th IEEE international requirements engineering conference (RE), IEEE, Melbourne, pp 166–177
85. Rocky SK, Rahim LA, Ahmad R, Sarlan A (2021) Hierarchical permissioned blockchain and traceability of requirement changes. In: 2021 international conference on intelligent cybernetics technology & applications (ICICyTA), pp 144–148
86. Sawyer P, Gacitua R, Stone A (2007) Profiling and tracing stakeholder needs. in: b. paech, c. martell (eds.) innovations for requirement analysis. From Stakeholders' Needs to Formal Designs, Lecture Notes in Computer Science, Springer, Monterey, CA, pp 196–213
87. Schwarz H (2009) Towards a comprehensive traceability approach in the context of software maintenance. In: 2009 13th european conference on software maintenance and reengineering, IEEE, pp 339–342
88. Serrano M, do Prado Leite JCS (2011) A rich traceability model for social interactions. In: Proceedings of the 6th international workshop on traceability in emerging forms of software engineering, TEFSE '11, ACM, New York, NY, pp 63–66
89. Shukla V, Auriol G, Baron C (2012) Integrated requirement traceability, multiview modeling, and decision-making: A systems engineering approach for integrating processes and product. In: 2012 IEEE international systems conference SysCon 2012, pIEEE, Vancouver, BC, pp 1–5
90. Sneed HM, Seidl R (2013) Softwareevolution: Erhaltung und Fortschreibung bestehender Softwaresysteme. dpunkt. verlag
91. Souali K, Rahmaoui O, Ouzzif M (2016) An overview of traceability: Definitions and techniques. In: 2016 4th IEEE international colloquium on information science and technology (CiSt), pp 789–793
92. Souali K, Rahmaoui O, Ouzzif M (2017) An overview of traceability: towards a general multi-domain model. Adv Sci Technol Eng Syst J (ASTES) 2(3):356–361
93. Spall S (1998) Peer debriefing in qualitative research: emerging operational models. Qual Inq 4(2):280–292
94. Spanoudakis G, Zisman A (2005) Software traceability: a roadmap. In: Handbook of software engineering and knowledge engineering, WORLD SCIENTIFIC, pp 395–428
95. Spijkman T, Dalpiaz F, Brinkkemper S (2022) Back to the roots: Linking user stories to requirements elicitation conversations. In: 2022 IEEE 30th international requirements engineering conference (RE), IEEE, pp 281–287
96. for Standardization, I.O (2018) ISO/IEC/IEEE 29148:2018(en). ISO
97. Stone A, Sawyer P (2006) exposing tacit knowledge via pre-requirements tracing. In: 14th IEEE international requirements engineering conference (RE'06), IEEE, Minneapolis/St. Paul, MN, pp 353–354
98. Stone A, Sawyer P (2006) Identifying tacit knowledge-based requirements. IEE Proc Softw 153(6):211–218
99. Torkar R, Gorschek T, Feldt R, Raja UA, Kamran K (2009) Requirements traceability state-of-the-art: a systematic review and industry case study. IST J
100. Torkar R, Gorschek T, Feldt R, Svahnberg M, Uzair Akbar R, Kamran K (2012) Requirements traceability : a systematic review and industry case study. Int J Softw Eng Knowl Eng 22(3):385–433
101. Tufail H, Masood MF, Zeb B, Azam F, Anwar MW (2017) A systematic review of requirement traceability techniques and tools. In: 2017 2nd international conference on system reliability and safety (ICSRS). IEEE, Milan
102. Urrego-Giraldo G (2003) Agent-based knowledge keep tracking. In: Proceedings fifth IEEE workshop on mobile computing systems and applications, IEEE, pp 8–16
103. Van Rompaey B, Demeyer S (2009) Establishing traceability links between unit test cases and units under test. In: 2009 13th European conference on software maintenance and reengineering, IEEE, pp 209–218
104. Weber-Jahnke JH, Onabajo A (2009) Finding defects in natural language confidentiality requirements. In: 2009 17th IEEE international requirements engineering conference, IEEE, Atlanta, GA, pp 213–222
105. Webster J, Watson RT (2002) Analyzing the past to prepare for the future: writing a literature review. Manag Inform Syst Res Center 26(2):13–23
106. Wibowo A, Davis J (2020) Requirements traceability ontology to support requirements management. In: Proceedings of the Australasian computer science week multiconference, ACSW '20, Association for computing machinery, New York, NY, pp 1–9
107. Wiegers KE, Beatty J (2013) Software requirements. Microsoft Press, Redmond
108. Winkler S, von Pilgrim J (2010) A survey of traceability in requirements engineering and model-driven development. Softw Syst Model 9(4):529–565
109. Wohlrab R, Steghöfer JP, Knauss E, Maro S, Anjorin A (2016) Collaborative traceability management: challenges and opportunities. In: 2016 IEEE 24th international requirements engineering conference (RE), pp 216–225
110. Wood D, Christel M, Stevens S (1994) A multimedia approach to requirements capture and modeling. In: Proceedings of IEEE international conference on requirements engineering, IEEE, Colorado Springs, CO, pp 53–56
111. Zhou X, Jin Y, Zhang H, Li S, Huang X (2016) A map of threats to validity of systematic literature reviews in software engineering. In: 2016 23rd Asia-Pacific software engineering conference (APSEC), pp 153–160