Framework for Requirements Traceability
- TLFRT supporting pre-RS & post-RS traceability

Uzair Akbar Raja and Kashif Kamran
This thesis is submitted to the School of Engineering at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Software Engineering. The thesis is equivalent to 40 weeks of full time studies.

Contact Information:
Author(s):
Uzair Akbar Raja
Address: Folkparksvägen 14:36, SE-37240 Ronneby, Sweden
Email: uzair_akbar@yahoo.com

Kashif Kamran
Address: Folkparksvägen 16:13, SE-37240 Ronneby, Sweden
Email: kashee79@hotmail.com

University advisor(s):
Dr. Tony Gorschek
School of Engineering

School of Engineering
Blekinge Institute of Technology
Box 520
SE – 372 25 Ronneby
Sweden

Internet : www.bth.se/tek
Phone : +46 457 38 50 00
Fax : + 46 457 271 25
Requirements traceability provides support for many software engineering activities like impact analysis, requirements validation and regression testing. In addition requirements traceability is the recognized component of many software process improvement initiatives. Requirements traceability also helps to control and manage evolution of a software system.

This thesis presents a systematic review and a framework for requirements traceability. The systematic review is aimed at presenting fair evaluation of research concerning requirements traceability over the period 1997 – 2007. The systematic review aims at identifying probable gaps in research about requirements traceability and opens new horizons to explore.

Moreover, two companies have been interviewed to understand the practice of requirements traceability in industry. After the analysis of industrial interviews these companies have been classified into two categories of traceability users. These categories are high-end traceability users and low-end traceability users.

Based on the analysis of systematic review results and industrial interviews, this thesis presents a framework for requirements traceability called ‘Three Level Framework for Requirements Traceability (TLFRT)’. This framework is composed of three levels. The level 0 of this framework focuses on pre-RS traceability where as level 1 and level 2 focuses on post-RS traceability. The level 1 provide traceability for the functional requirements, where as level 2 provide traceability for non-functional requirements. TLFRT provides guidelines to a process that can be tailored to fit the needs of the high-end traceability users and low-end traceability users as well.

This framework has been statically validated in two companies. The validation of TLFRT in industry resulted in compilation of lessons learned, which ensures that this theoretical framework could also be used in real industrial environment. The results of the research presented in this thesis are aimed at supporting requirements traceability by taking its current issues/challenges into account.

Keywords: Requirements traceability, Framework, Systematic review.
ACKNOWLEDGEMENTS

We would like to extend our sincere gratitude to our supervisor Dr. Tony Gorschek. His guidance, worthy criticism and appreciation helped us to achieve our goals. Dr. Tony Gorschek not only guided us in every crucial step of our thesis work but also provided support in all respect. We really admire his ability of always being there, despite his busy schedule. We have yet to learn half of what he knows.

We are grateful to Kent Pettersson, Librarian at Blekinge Tekniska Högskola for helping us to define search terms and identify search resources in the review protocol. We recognize his efforts to conduct pilot searches and identifying the relevance of search terms.

We appreciate the efforts and positive feedback of Wasif Afzal (PhD student) and Sebastian Barney (PhD student) for evaluating our review protocol.

We would like to express our gratitude to Ewa Kristoffersson, Therese O-Starheim of Ericsson AB Karlskrona and Kennet Henningsson of UIQ Technolog Ronneby Sweden. They have been very kind and cooperative for providing us time for our industrial interviews part and framework validation. Without them the validation of our framework has never been possible and this thesis would have looked completely different. We are indebted to our most encouraging colleague Muhammad Nadeem Khan of Ericsson AB Karlskrona Sweden. He helped us a lot in providing industrial contacts for our thesis.

Colleagues and friends at Blekinge Tekniska Högskola also deserve thanks. Instead of mentioning them with names and accidentally leaving some of them out, we want to appreciate all the encouragement and nice memories. However, a few of our friends have contributed more than others. We would like to admire the efforts of our senior colleague Shahid Mahmood and our German Friend Martin Baumer for sparing their time to proofread the thesis. We would also like to thank our friends Sheraz Ahmad, Farrakhir Saeed, Syed Abdul Basit and Choudhary Aleem Gujjar for cheering us throughout the thesis tenure.

We would also like to thank our faculty reviewer Dr. Robert Feldt for providing us useful feedback on our thesis proposal. We have developed confidence in our work due to his idea of validating the framework in industry.

Our special thanks are for our parents, brothers and sisters for scarifying their wonderful time by sending us abroad in the quest of knowledge. They are the best.
CONTENTS

ABSTRACT ............................................................................................................................................... III

ACKNOWLEDGEMENTS ........................................................................................................................ IV

CONTENTS ................................................................................................................................................ V

TABLE OF FIGURES ............................................................................................................................. VIII

TABLE OF TABLES .................................................................................................................................... IX

1 INTRODUCTION ................................................................................................................................... 1
  1.1 AIMS AND OBJECTIVES .................................................................................................................... 4
  1.2 RESEARCH QUESTIONS ...................................................................................................................... 4
  1.3 EXPECTED OUTCOMES .................................................................................................................... 5
  1.4 RESEARCH METHODOLOGY ........................................................................................................... 5
  1.5 RELATED WORK ................................................................................................................................ 6
  1.6 STRUCTURE OF THE THESIS ........................................................................................................... 7

2 SYSTEMATIC REVIEW .......................................................................................................................... 9
  2.1 PLANNING THE REVIEW .................................................................................................................... 9
    2.1.1 The Need for a Systematic Review ............................................................................................... 9
    2.1.2 Development of a Review Protocol .............................................................................................. 9
    2.1.3 Evaluating a Review Protocol .................................................................................................... 13
  2.2 CONDUCTING THE REVIEW ............................................................................................................ 14
    2.2.1 Identification of Research ......................................................................................................... 14
    2.2.2 Selection of Primary Studies ..................................................................................................... 14
    2.2.3 Study Quality Assessment ........................................................................................................ 17
    2.2.4 Data Extraction & Monitoring .................................................................................................. 17
    2.2.5 Data Synthesis ........................................................................................................................... 17
  2.3 REPORTING THE REVIEW ............................................................................................................... 18

3 SYSTEMATIC REVIEW RESULTS ......................................................................................................... 19
  3.1 DEFINITIONS OF REQUIREMENTS TRACEABILITY ........................................................................ 19
    3.1.1 Analysis ..................................................................................................................................... 21
  3.2 CHALLENGES/ISSUES IN IMPLEMENTATION OF REQUIREMENTS TRACEABILITY ..................... 22
    3.2.1 Analysis ..................................................................................................................................... 25
  3.3 REQUIREMENTS TRACEABILITY TOOLS ..................................................................................... 27
    3.3.1 RETRO ..................................................................................................................................... 27
    3.3.2 Rational Requisite Pro ............................................................................................................... 27
    3.3.3 DOORS ..................................................................................................................................... 27
    3.3.4 DesignTrack ............................................................................................................................... 27
    3.3.5 TRAM ....................................................................................................................................... 28
    3.3.6 Scenario Advisor Tool ............................................................................................................... 28
    3.3.7 Other Traceability Tools ............................................................................................................ 28
    3.3.8 Analysis ..................................................................................................................................... 29
  3.4 REQUIREMENTS TRACEABILITY TECHNIQUES ............................................................................. 30
    3.4.1 Value Based Requirements Traceability (VBRT) ....................................................................... 30
    3.4.2 Feature Oriented Requirements Tracing (FORT) ....................................................................... 32
    3.4.3 Pre-RS Requirements Tracing .................................................................................................... 35
    3.4.4 Event Based Traceability (EBT) ................................................................................................. 38
    3.4.5 Information Retrieval (IR) ........................................................................................................ 39
    3.4.6 Rule Based (RB) Approach ....................................................................................................... 39
    3.4.7 Hyper-text Based Approach (HB) ............................................................................................ 39
    3.4.8 Feature-Model Based Approach (FB) ....................................................................................... 39
    3.4.9 Scenario-Based Approach (SB) ................................................................................................. 40
    3.4.10 Process Centered Environments .............................................................................................. 40
3.4.11 Design Patterns........................................................................................................40
3.4.12 Traceability Matrices..................................................................................................40
3.4.13 Keywords and Ontology.............................................................................................40
3.4.14 Aspect Weaving.........................................................................................................40
3.4.15 Goal Centric Traceability (GCT)...............................................................................41
3.4.16 Analysis.......................................................................................................................42
3.5 ANSWERING THE RESEARCH QUESTIONS STATED IN REVIEW PROTOCOL.........46
4 INDUSTRIAL INTERVIEWS...............................................................................................49
4.1 COMPANY A..................................................................................................................49
4.1.1 Introduction................................................................................................................49
4.1.2 Interviewee................................................................................................................49
4.1.3 Requirements Traceability..........................................................................................49
4.1.4 Tool Support for Traceability.....................................................................................49
4.1.5 Factors Influencing Requirements Traceability.........................................................50
4.1.6 Future Considerations...............................................................................................50
4.2 COMPANY B..................................................................................................................50
4.2.1 Introduction................................................................................................................50
4.2.2 Interviewee................................................................................................................50
4.2.3 Traceability Practices.................................................................................................50
4.2.4 Tool Support for Traceability.....................................................................................51
4.2.5 Benefits of Implementing Traceability.......................................................................51
4.2.6 Factors Influencing Requirements Traceability.........................................................51
4.3 ANALYSIS OF INTERVIEWS........................................................................................52
5 A FRAMEWORK FOR REQUIREMENTS TRACEABILITY (TLFRT)...............................56
5.1 LEVEL 0: PRE-RS TRACEABILITY ..............................................................................57
5.1.1 Example....................................................................................................................58
5.1.2 Issues solved by Level 0............................................................................................61
5.1.3 Limitations of Level 0...............................................................................................61
5.1.4 Rationale for using Pre-RS Requirements Tracing.....................................................62
5.2 LEVEL 1: POST-RS TRACEABILITY WITH FUNCTIONAL REQUIREMENTS.......62
5.2.1 Requirements definition.............................................................................................62
5.2.2 Requirements prioritization.......................................................................................62
5.2.3 Requirements packaging...........................................................................................64
5.2.4 Requirements linking...............................................................................................65
5.2.5 Evaluation................................................................................................................65
5.2.6 Issues Solved by Level 1..........................................................................................66
5.2.7 Limitations of Level 1...............................................................................................66
5.2.8 Rationale for using VBRT........................................................................................66
5.3 LEVEL 2: POST-RS TRACEABILITY WITH NON-FUNCTIONAL REQUIREMENTS..67
5.3.1 Goal Modelling..........................................................................................................67
5.3.2 Impact Detection........................................................................................................69
5.3.3 Goal Analysis............................................................................................................70
5.3.4 Decision Making.......................................................................................................70
5.3.5 Issues Solved by Level 2..........................................................................................71
5.3.6 Limitations of Level 2...............................................................................................71
5.3.7 Rationale for Using GCT..........................................................................................71
6 STATIC VALIDATION OF THE FRAMEWORK.........................................................72
6.1.1 Validation Design.......................................................................................................72
6.1.2 Validation Feedback from Company ’A’.....................................................................73
6.1.3 Validation Feedback from Company ’B’.....................................................................73
6.1.4 Lesson Learned..........................................................................................................74
6.2 ANSWERING THE RESEARCH QUESTIONS...............................................................75
7 VALIDITY THREATS........................................................................................................77
7.1 CONCLUSION VALIDITY (RELIABILITY).................................................................77
7.2 CONSTRUCT VALIDITY.................................................................................................77
7.3 INTERNAL VALIDITY (CAUSALITY)............................................................................78
TABLE OF FIGURES

Figure 1: Requirement Traceability with in Software Engineering .............................................. 2
Figure 2: Different Types of Traceability [10] .................................................................................. 3
Figure 3: Work Flow in Relation with Research Questions ............................................................. 6
Figure 4: Structure of the Thesis ..................................................................................................... 8
Figure 5: Year-wise publication of articles on requirements traceability during 1997-2007 ........... 17
Figure 6: Requirements Traceability Definition Statistics ............................................................... 22
Figure 7: Overview of VBRT process [21] ...................................................................................... 31
Figure 8: Feature Modeling [28] .................................................................................................... 33
Figure 9: Overview of Feature Oriented Requirements Tracing [28] ............................................. 34
Figure 10: Capabilities Engineering Process [30] ......................................................................... 36
Figure 11: Functional decomposition (FD) graph [30] ................................................................. 37
Figure 12: Event Based Traceability [53] ....................................................................................... 38
Figure 13: Overview of GCT process [24] .................................................................................... 42
Figure 14: Statistical representation of the empirical evidences of requirements traceability techniques ........................................................ 44
Figure 15: Statistical representation of number of papers presenting the same technique ......... 44
Figure 16: Requirements Traceability Issues Based on Systematic Review ............................... 47
Figure 17: Requirements Traceability Tools Based on Systematic Review ................................. 47
Figure 18: Requirements Traceability Techniques Based on Systematic Review ..................... 48
Figure 19: Requirements Traceability in Company B ................................................................... 51
Figure 20: Three Level Framework for Requirements Traceability (TLFRT) ............................... 56
Figure 21: Pre-RS Requirements Tracing (Level 0) ....................................................................... 57
Figure 22: Requirements Abstraction Model (RAM) [13] ............................................................. 61
Figure 23: Traceability between Various Software Artefacts Using RTM ..................................... 65
Figure 24: Goal Modelling of ‘Performance of local system’ NFR using SIG ......................... 68
Figure 25: Establishing Traceability Links Between Functional Model and SIG Elements ....... 69
Figure 26: Potentially Impacted Elements in SIG due to Functional Change ......................... 70
Figure 27: Overview of the Methodology for Answering Research Questions ............................ 76
TABLE OF TABLES

Table 1: List of selected journals and conferences .......................................................... 11
Table 2: Articles selected for systematic review ............................................................... 15
Table 3: Articles on R.T in selected journals and conferences ....................................... 16
Table 4: Articles defining requirements traceability ....................................................... 20
Table 5: Overview of Requirements Traceability Definitions ........................................ 21
Table 6: Articles reporting issues related to requirements traceability ........................... 24
Table 7: Issues Related to Requirements Traceability Addressed by Academia .............. 26
Table 8: Issues Related to Requirements Traceability Addressed by Industry ............... 26
Table 9: Issues Related to Requirements Traceability Un-solved in Academia & Industry .... 26
Table 10: Articles reporting traceability tools ............................................................... 28
Table 11: Comparison of traceability tools ..................................................................... 29
Table 12: Steps and activities for feature modelling [28] ................................................. 32
Table 13: Priority levels and classification of artefacts [28] ............................................ 34
Table 14: Results of case study on car rental system [28] .............................................. 35
Table 15: Comparison of VBRT and FORT ................................................................... 35
Table 16: Pre-RS traceability using capabilities engineering approach .......................... 38
Table 17: Comparison of Traceability Techniques ............................................................ 43
Table 18: Conclusion drawn from the Empirical Evidence of Traceability Techniques .... 45
Table 19: Traceability issues in Company A & Company B ............................................ 53
Table 20: Views of Company A & B about the issues identified in systematic review .... 53
Table 21: Most Important Problems/Issues Deduced From Systematic Review and Industrial Interviews ......................................................................................... 55
Table 22: Issues for the Framework ................................................................................ 55
Table 23: User and System Views in Level 0 ................................................................. 58
Table 24: User and System Views at Level 0 of Library Information System ................... 60
Table 25: Functional Requirements in Library Information System ............................... 60
Table 26: NFR in Library Information System ................................................................. 61
Table 27: Requirements Prioritized By the Project Manager/Product Manager (PM) ........ 63
Table 28: Requirements Prioritized By the Stakeholders (S) .......................................... 63
Table 29: Prioritized Requirements with Priority Levels .................................................. 64
Table 30: Comparison of Tracing Techniques [21] ......................................................... 67
Table 31: Notations used for SIG .................................................................................. 68
Table 32: Questionnaire for Framework Validation ....................................................... 72
1 INTRODUCTION

According to Sommerville & Sawyer [18] requirement engineering involves the activities for discovering, documenting and maintaining a set of requirements for a computer-based system. Requirement engineering activities are often divided into five categories. These categories include requirement elicitation, requirement analysis, requirement specification, requirement validation and requirement management. These categories are briefly defined below [18, 19, 66].

Requirements elicitation is the process through which the customer and requirement engineers discover, articulate, review, and verify user requirements. Various techniques like interview, surveys, questionnaires, observation and Joint Application Design (JAD) workshops are used to elicit requirements from stakeholders.

Requirements analysis is the process through which conflicts, omissions, overlaps and inconsistencies are detected and then resolved with the participation of stakeholders. Requirement prioritization is an implicit part of requirement analysis.

Requirements specification is the process through which all the agreed requirements are properly documented. Requirements need to be documented in a consistent way by using standard conventions. This document is called as Software Requirement Specification (SRS). It is formalized to act as a basis for contractual purposes between the customer and the developer or supplier of the software.

Requirement validation/verification ensures that requirements in the software requirements specification are complete and consistent. After developing requirement specification some validation and verification procedures are applied to ensure requirements understandability on the part of the software developer. Requirements’ correctness like completeness and consistency and feasibility properties like required cost and required resources are evaluated and analyzed.

Requirement management assists in maintaining the requirements’ evolution throughout the development project. According to Sommerville and Sawyer requirement management is concerned with all of the processes involved in changing system requirements. The principal concerns of requirement management are:

1. Managing changes to agreed requirements
2. Managing the relationships between requirements
3. Managing dependencies between the requirement document and other documents produced during the systems and software engineering process.

According to Gorschek [13] documentation, change management and traceability are the key activities of requirement management. One of the main tasks of requirement management is to assure requirements traceability from start through the system evolution and maintenance. In addition to this traceability is desired in a system where changes occur so frequently. Traceability is also recommended as a necessary activity by various standards such as IEEE Std. 830-1998 and CMMI. According to Easterbrook & Nuseibeh [67] requirements traceability lies at the heart of requirement management.
Figure 1: Requirement Traceability with in Software Engineering

Figure 1 depicts the place of requirement traceability within software engineering (SE), requirement engineering (RE), and requirement management (RM).

There is a myth that if you cannot trace you cannot manage [10]. To manage a system effectively one has to take care of all aspects of the system. Requirements traceability is an important activity within requirements management.

According to Gotel & Finkelstein [11] requirements traceability is “the ability to describe and follow the life of a requirement in both forwards and backwards direction (i.e., from its origins, through its development and specification to its, subsequent deployment and use, and through periods of on-going refinement and iteration in any of these phases)”.

According to Gorschek et al [13] Backward traceability indicates the link from requirement to its origin; this origin can be a person, document or group of persons. Forward traceability indicates links from requirements to all artefacts of software development life cycle.

The definition of requirement traceability by Gotel & Finkelstein [11] is considered as comprehensive definition that’s why researcher like Gorschek [13], Jane Cleland [6], Aurum and Wohlin [10] refer the same definition. The same definition is referred by software engineering institute official website [12]. SWEBOK [15], defines requirements tracing as, “Requirements tracing is concerned with recovering the source of requirements and predicting the effects of requirements”.

Another definition of requirement traceability is given by Palmer [7] as “traceability gives essential assistance in understanding the relationships that exist within and across software requirements, design, and implementation”. These kinds of relationship help the project manager, requirement engineers, designers and all other stockholders in their concerned phases.

According to Gotel and Finkelstein [1] there are two aspects of requirement traceability, Pre-RS traceability and Post-RS traceability. Pre-RS traceability is concerned with those aspects of requirement’s life from the point where they are not included in the requirements specification [10]. In Pre-RS traceability requirements are related to their origin and other requirements. The origin of requirements includes stakeholders(S), business rule (BR) or previous documents (Doc) as shown in Figure 2. In Figure 2 requirement R2.1 is related to requirements R1 and R2. Post-RS traceability is concerned with those aspects of a requirement’s life from the point where it is included in the requirements specification and forward [10]. It ensures that all the requirements are fulfilled by system design and implementation. In Post-RS traceability requirements are related to test cases which ensure that components satisfy those requirements. Besides these two aspects of requirements traceability Davis et al. has classified traceability into the following four types [19].

1. **Backward-from traceability**
   - It links requirements to their sources which are in other documents or from people.

2. **Forward-from traceability**
It links requirements to the design and implementation components.

3. **Backward-to traceability**
   It links design and implementation components back to the requirements.

4. **Forward-to traceability**
   It links other documents to relevant requirements. These other documents can be for example operation manuals, describing the system functionality.

---

**Figure 2: Different Types of Traceability [10]**

In short requirement traceability is an important aspect of requirements management, where requirements are evidently connected to their origin and to the other software artefacts. Initially requirement traceability was used in the development of safety critical systems only. Nowadays it is in use in the development of other systems as well. Widely accepted standards such as ISO 9000 and CMMI require traceability practices to be implemented [6, 8]. Requirements traceability plays a vital role in software evolution and maintenance in addition to development [9]. Requirements traceability can prove to be beneficial in following ways

1. Requirement Traceability helps to identify the source of the requirement either it is issued by a person or document or group of persons [22].
2. It helps in performing impact analysis [4, 15, 65] which traces out what other component(s) might be affected in response to change in a particular requirement.
3. Requirement Traceability helps in test case verification for example which test cases verify which requirement [22].
4. It helps tracking the overall progress of the project for example we can measure how much requirements are implemented, how much requirements are in design phase and how much requirements are completed.

It is a common observation that requirements are interdependent on each other and on other artefacts. Well established rules of traceability helps to trace the relationship between requirements and with other artefacts. For instance in Figure 2 requirement R1, R2 and R2.1 are interdependent.

In spite of the above mentioned benefits companies are reluctant to implement traceability practices. One main aspect which might obstruct the companies from having traceability practices is the associated time and effort. For example the development of requirement traceability matrix (RTM) is time consuming, laborious and error prone [6]. These factors obstruct the implementation of requirement traceability. On the basis of these factors organizations consider traceability costly over benefits. In one of their study Gotel & Finkelstein [1] found that due to tight deadlines, less
communication between developers, lack of tools support and general perception about cost and benefits to develop RTM, effective implementation of requirement traceability is not possible.

In another study by Ramesh [7], found that organizations involve “low end users” in requirement traceability practices. These end users are not capable to get maximum benefit of their traceability efforts due to lack of sufficient training. Therefore, they can only create some basic links between artefacts but unable to get benefit from higher level of automation which will increase the benefits of their traceability efforts [7]. In addition to it traceability using automated tool can also be costly for an organization, because these tools also involve human analyst for evaluation and filtration of the results [6].

In current era organizations recognize the importance of requirement traceability. According to [4, 5] requirement traceability supports various critical activates through the whole software development life cycle such as requirement validation, regression test case selection, compliance verification and impact analysis.

Although organizations are aware of the importance of requirement traceability but they are not willing to implement it because they think in terms of return on investment. We aim to investigate the major benefits, and potential pitfalls regarding requirements traceability. All this is an attempt to establish the state-of-the-art of requirements traceability as well as to identify the main challenges reported by researchers and industry. In order to achieve this we will do systematic review and interviews with the industry personals. The systematic review is never done by any researcher in the field of requirement traceability, the motivations for choosing systematic review as methodology can be find in Section 1.4 & 2. In compliance with this we intend to develop a framework which takes the identified challenges into account. Then this framework will be validated by the same industry personals to determine whether it is helpful to resolve their traceability related issues.

The proposed framework is based on the results from two sources such as systematic review and industrial interviews. Up to the best of our knowledge there is no framework on requirements traceability which is based on such kinds of inputs.

1.1 Aims and Objectives

The main aim of this thesis is to develop a framework for requirements traceability. Following objectives are set in order to achieve this aim.

- Identify the current state of research in requirements traceability using a systematic review.
- Identify the factors that obstruct the implementation of traceability practices based on reports from academia and industry.
- Investigate if any observable differences can be found in relation to the factors reported from academia versus industry experience reports.
- Combine various traceability techniques to develop a framework for requirements traceability based on systematic review and interviews in software companies.

1.2 Research Questions

Following research questions will be answered during the thesis.

RQ1. What is requirements traceability based on the state-of-the-art in research and standards?
The up to date research on requirements traceability will be carried out with the help of systematic review. The focus of systematic review will be on the requirements traceability over the decade. This will help to understand various dimensions of requirements traceability, requirements traceability tools and techniques.

RQ2. What are the main factors that are reported by the academia and industry as impeding proper implementation of requirements traceability?
Some of the factors for instance lack of coordination between people, failure to follow standards, requirements traceability costs obstruct the implementation of requirements
traceability. The impeding factors reported in academia will be revealed by the systematic review. Interviews will be conducted in software companies to find out the impeding factors from industry.

**RQ3.** Is it possible to develop a generic traceability framework that addresses the issues identified in RQ2? RQ3 can be seen as directly related to RQ2. A generic traceability framework will be proposed to solve the traceability obstruction factors identified by RQ2.

**RQ4.** Does the proposed framework address the problems faced by the industry? This question is related to RQ3. The proposed framework will be presented to the industry personnel that were previously interviewed. This will provide the feedback from practitioners.

Following are the deliverables against each research question.

- **D1.RQ1.** A definition of requirements traceability based on research.
- **D1.RQ2.** Set of qualitative and quantitative questions designed for interview to figure out requirement traceability problems in two companies.
- **D2.RQ2.** List of challenges/problems affecting implementation of requirements traceability.
- **D3.RQ2.** Report on complete systematic review. The RQ1 and RQ2 will be answered with the help of a systematic review. In case of RQ2 the systematic review will provide the factors affecting requirements traceability as reported by the academia and industry. Similarly the factors affecting requirements traceability practices reported by the selected companies will be provided by the industrial interviews.
- **D1.RQ3.** Developed framework of requirements traceability based on the systematic review.
- **D2.RQ3.** Set of qualitative and quantitative questions designed for interview to validate the proposed framework from interviewees.
- **D1.RQ4.** Framework validity report derived from interviews from two software companies.
- **D2.RQ4.** Final draft of thesis report.

### 1.3 Expected Outcomes

The expected outcome will be a report providing current state of research in requirements traceability. One of our main outcomes will be a systematic review. On the basis of knowledge gained from systematic review and input from two software related companies, we intend to suggest a framework for requirements traceability. After this we will validate the framework from two software related companies.

### 1.4 Research Methodology

A mixed approach using both quantitative and qualitative research methodologies was adopted. The interviews were conducted with the software companies to identify requirements traceability problems. In addition to this a systematic review was also carried out.

Systematic review and industrial input provided better understanding of the current state-of-research in the field of requirement traceability. The systematic review was performed by following the guidelines given by Kitchenham report [16]. After performing analysis of systematic review and industrial interviews a framework for requirements traceability was proposed. In order to validate our purposed framework we conducted semi-structured interviews with the selected companies. The rationales behind doing interviews are as follows.

1. To figure out which of the main challenges as are identified in the review, the companies feel relevant to them.
2. To make sure that the stakeholders understand the developed framework.
3. To trace out whether the framework provides help to meet the challenges.

There are three types of interviews, structured, unstructured and semi-structured. In structured interviews the questions should be arranged in “if-then” structure so that they can lead the discussion in the specific direction based on the answers of previous questions [17]. In unstructured interviews, interviewee is the source of questions and answers [17]. During unstructured interview, interviewer can use a short list of topics to be touched upon by open ended questions [17]. In semi-structured interviews, a combination of both structured and unstructured interviews are used [17].

The reason to conduct semi-structured interviews is because they comprise both open ended and specific questions. This will help to conduct the interview in comprehensive manner.

![Figure 3: Work Flow in Relation with Research Questions](image)

1.5 Related Work

The preliminary study of literature indicates that no systematic review has been published on requirements traceability. Similarly there are some frameworks related to requirements traceability reported in literature. However, some of these frameworks have been validated in industry. But there is no evidence that these validated frameworks are used in industry. A brief introduction of these frameworks is given below.

A heterogeneous traceability framework called TraCS (Traceability for Complex Systems) has been proposed by J. Cleland-Huang et.al. [5]. TraCS provides extensible framework for integrating heterogeneous traceability techniques. The prototype of TraCS contains well defined user interface that provides transparency to enable users to issue traceability queries without knowing about the underlying traceability mechanism. But there is no empirical proof of the validation of TraCS in industry. In future work it is proposed that TraCS will be implemented in the form of a fully extensible system.

A framework of traceability called “Traceability meta-model” has been proposed by Ramesh et.al. in [39]. This model represents different traceability linkages between various objects produced during system life cycle. In addition it describes traceability models at various levels of abstraction, so that in practice different organizations may implement components of traceability at different levels of detail. This framework has been implemented as a pilot study in a flight control system of a government organization. The case study results suggest that the cost associated with traceability can be justified in terms of producing better quality product, lower life cycle and maintenance costs.

In [43] XTraQue tool has been proposed by W. Jirapanthong and A. Zisman. The XTraQue tool contains a sophisticated user interface in which user can select documents to define traceability relationships. The XTraQue tool is based on rule based traceability. It allows creation of new traceability rules and execution of these rules to verify their correctness. When a user is satisfied with
a new rule, it can be inserted in the document containing the traceability rules. The precision and accuracy of this tool has been verified by an experiment.

In [51] a framework called Traceable Development Contract (TDC) has been proposed by P. Arkley and S. Riddle for recording traceability information. TDC provides a means of controlling the interaction of development teams and recording the traceability relationships beneficial to the development process.

In [54] a model for traceability has been proposed by S. Ahmed. This model is composed of Traceability Engine Component (TEC), a Traceability Viewer Component (TVC) and Quality Assurance Interface (QAI). The TEC helps developers to correlate source code with elements of software requirements. TVC facilitates viewing correlation between requirements and code. The component QAI addresses the validation and verification of requirements. The TEC, TVC and QAI provide very efficient way of tracking and tracing requirements [54]. But there is no empirical proof of validation of this model in industry. In addition in the future work it is stated that this model will be implemented in a tool.

In article [59] a framework for dealing with media has been proposed by O. Gotel and S.J. Morris. In the area of requirements traceability this framework helps requirements engineers to make informed decisions about media choices and transformations when crafting requirements records. But there is no empirical evidence of validation of this framework in industry.

In article [70] a framework for managing traceability relationships between requirements and architectures has been proposed by S.A. Sherba and K.M. Anderson. This framework is based on the techniques from open hypermedia and information integration. Open hypermedia systems enable the creation and viewing of relationships in heterogeneous applications. Where as information integration provide services to automate discovery, creation and maintenance of relationships between heterogeneous artefacts. This framework has been implemented in a prototype. But there is no empirical evidence of validation of this framework in industry.

1.6 Structure of the Thesis

This section describes the overall structure and contents of the thesis.

Chapter 2 (Systematic Review) provides a detailed discussion about the basic elements of the systematic review. It contains review protocol which contains a comprehensive plan for conducting a systematic review. Review protocol contains search terms, names of databases to be searched, inclusion/exclusion criteria, study quality assessment procedure and the way to synthesize the data gathered from the systematic review. The process of systematic review on requirements traceability is completely documented in this chapter.

Chapter 3 (Systematic Review Results) provides the results of the systematic review. This chapter contributes to answering the research questions proposed in the review protocol.

Chapter 4 (Industrial Interviews) contains the results of the interview conducted in two software development companies. These companies were interviewed to explore the traceability practices carried out in industry and issues related to traceability.

Chapter 5 (Framework for Requirements Traceability) contains the proposed framework on requirements traceability. This framework is developed based on the issues identified by the systematic reviews and industrial interviews. In addition it contains the validation of the framework by the practitioners previously interviewed in selected companies.

Chapter 6 (Static Validation of Framework) reports the validation of the framework by the practitioners previously interviewed in selected companies.

Chapter 7 (Validity Threats) reports the validity threats related to the overall research design. Validity threats add value to the research study by identifying those factors which can affect the results.

Chapter 8 (Epilogue) provides conclusions drawn from this study. It also contains future work and identifies new dimensions in the field of requirements traceability.
Chapter 2
Systematic Review

The chapter provides a detailed discussion about the basic elements of the systematic review. It also contains review protocol which contains a comprehensive plan for conducting a systematic review.

Chapter 3
Systematic Review Results

This chapter provides the results of the systematic review. This chapter contributes to answering the research questions proposed in the review protocol.

Chapter 4
Industrial Interviews

This chapter contains the results of the interview conducted in two software development companies.

Chapter 5
Framework for Requirements Traceability

This chapter contains the proposed framework on requirements traceability. In addition it contains an example of a library information system, which is traced with the help of proposed framework.

Chapter 6
Static Validation of Framework

This chapter contains the validation of the framework by the practitioners previously interviewed in selected companies.

Chapter 7
Validity Threats

This chapter reports the validity threats related to the overall research design. Validity threats add value to the research study by identifying those factors which can affect the results.

Chapter 8
Epilogue

This chapter provides conclusions drawn form this study. It also contains future work and identifies new dimensions in the field of requirements traceability.

Figure 4: Structure of the Thesis
2 **SYSTEMATIC REVIEW**

A systematic literature review provides a mechanism for evaluating, identifying and interpreting all available research relevant to a particular research question, topic area or phenomenon of interest [16]. Individual studies contributing to the systematic review are known as primary studies while a systematic review is a secondary study.

A systematic literature review comprises three main phases namely planning the review, conducting the review and reporting the review. In the planning phase need for the systematic literature review is identified and a review protocol is made. This review protocol serves as a comprehensive search guide throughout the systematic literature review. Conducting the literature review involves identification of research, primary studies selection, study quality assessment, data extraction & monitoring and data synthesis. The phase of reporting the systematic literature review is a single stage phase.

The stages involved in the systematic literature review seem to be sequential but actually they may involve a number of iterations [16]. For example, many activities in the review protocol like search terms, inclusion and exclusion criteria for primary studies are revised when actually the review is conducted. Kitchenham et al. [16] identifies some of the features that distinguishes systematic literature reviews from conventional literature reviews as,

- Systematic reviews start by defining a review protocol that specifies the research questions, search terms, resources to be searched during the systematic review.
- The search strategy defined in the review protocol aims to detect as much of the literature as possible.
- Systematic reviews document their search strategy so that the readers can access its rigor and completeness.
- Systematic reviews require explicit inclusion and exclusion criteria for each primary study.
- Systematic reviews specify the quality criteria to evaluate each primary study and data extraction forms to extract the information from each primary study.
- Systematic reviews are pre-requisite for quantitative meta-analysis.

2.1 **Planning the Review**

2.1.1 **The Need for a Systematic Review**

The objective of this systematic review is to summarize the work done in requirements traceability during the years 1997-2007. This systematic review will provide a list of reported problems in requirements traceability and various requirements traceability techniques. These techniques will be helpful in designing the framework for requirements traceability.

2.1.2 **Development of a Review Protocol**

The review protocol is a detailed plan for conducting the systematic review specifying the methods for selecting the primary studies. Review protocol reduces the possibility of researcher bias [16].

2.1.2.1 **Background**

Systematic reviews are mostly used in the field of medicines but nowadays they are also used in software engineering research supporting Evidence-based Software Engineering (EBSE) [16]. The aim of EBSE, as described by Dybå et al. is, “To provide the means by which current best evidence from research can be integrated with practical experience and human values in the decision making process regarding the development and maintenance of software” [20].
The review protocol serves as the backbone of the systematic review. It contains research questions, search terms, names of resources to be searched, study inclusion/exclusion criteria, study quality assessment checklists, data extraction forms and extracted data synthesis strategy.

The purpose of the study described in this review protocol is to review the current status of research in requirements traceability from 01 Jan, 1997 to 30 Sep, 2007. The systematic review of requirements traceability will provide an overview of the area. This systematic review will serve as an input in the development of generic framework for requirements traceability.

2.1.2.1.2 Research Questions
Following research questions will be answered during the systematic review.
SRQ 1. What is requirements traceability based on state-of-the-art research?
SRQ 2. What are the challenges/problems in implementation of requirements traceability and how did research address these problems?
SRQ 3. What are the various requirements traceability tools?
SRQ 4. What are the various requirements traceability techniques?

2.1.2.1.3 Search Strategy
The search process will be online search using search terms and resources to be searched listed below.

a) Search terms
Following search terms will be used to extract the primary studies.
1. Requirements Traceability.
2. Requirements Traceability Technique(s).
3. Traceability.
4. Requirements Tracing.
5. 1 AND Challenges
6. 1 AND Problems
7. 1 AND Issues
8. 1 AND Success Stories
9. 2 AND Challenges
10. 2 AND Problems
11. 2 AND Issues
12. 2 AND Success Stories
13. 5 OR 6 OR 7 OR 8
14. 9 OR 10 OR 11 OR 12
15. 1 AND Experience Reports
16. 1 AND Process Improvement
17. 1 AND Impact
18. 1 AND Experiences
19. 1 AND lessons Learned
20. 1 AND Good Practices
21. 1 AND Standards
22. 1 AND Return on Investment
23. 1 AND ROI

b) Resources to be searched
Following online resources will be used during the systematic review.
- IEEE Xplorer
In addition to the online search following conferences and journals will be manually explored year-wise starting from Jan 1997 to Sep 2007, to reduce the probability of missing an important article.

Table 1: List of selected journals and conferences

<table>
<thead>
<tr>
<th>Sr No</th>
<th>Journal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IEEE Transactions on Software Engineering (TSE)</td>
</tr>
<tr>
<td>2</td>
<td>ACM Transaction on Software Engineering Methodology (TOSEM)</td>
</tr>
<tr>
<td>3</td>
<td>Springer Requirements Engineering Journal</td>
</tr>
<tr>
<td>4</td>
<td>Springer Innovations in Systems And Software Engineering</td>
</tr>
<tr>
<td>5</td>
<td>Springer Software and Systems Modeling</td>
</tr>
<tr>
<td>6</td>
<td>Springer Annals of Software Engineering</td>
</tr>
<tr>
<td></td>
<td><strong>Conference Proceedings</strong></td>
</tr>
<tr>
<td>1</td>
<td>ACM International Conference on Software Engineering (ICSE)</td>
</tr>
<tr>
<td>2</td>
<td>IEEE International Conference on Requirements Engineering</td>
</tr>
<tr>
<td>3</td>
<td>IEEE International Symposium on Requirements Engineering</td>
</tr>
</tbody>
</table>

2.1.2.1.4 Study Selection Criteria and Procedures

The inclusion and exclusion criteria will be as follows.

a) Study Inclusion Criteria

The articles on requirements traceability published between 01 Jan, 1997 and 30 Sep, 2007 will be included. The following criteria will help to judge the suitability of the article to be included in the systematic review.

1) The article should be a peer-reviewed article.
2) The article can be case study, experiment, survey, experience report, comparative evaluation or action research.
3) The article is included if it reports success/issues/failures or any type of experience of requirements traceability.
4) The article should be available as full text article.
5) The article should be based on research done in requirements traceability.
6) The article will be included if it contains definition of Requirements Traceability.
7) Articles introducing new and important claims regarding Requirements Traceability and supporting these claims with some sort of evidence.
8) The article is included if it identifies the problems / challenges in requirements traceability.
9) The article is included if it provides some sort of solution / roadmap / framework of traceability problems.
10) An article is included if it evaluates or compares two or more requirements traceability techniques.

b) Study Exclusion Criteria
The articles that does not qualify the study inclusion criteria will be excluded.

2.1.2.1.5 Study Selection Process
The study selection process will be based on reading of the title, abstract and conclusion of the particular research article. If it satisfies our inclusion criteria then whole article will be read.

2.1.2.1.6 Study Quality Assessment and Procedures
The selected research papers will be evaluated based on following criteria for research paper structure (Introduction, Method, Results, Analysis, Discussion / Conclusion).

Introduction
✓ Does the introduction provide the overview of requirements traceability?

Method
✓ Is the research methodology clearly defined in the research paper?
✓ Is the research methodology appropriate for problems under consideration?

Results
✓ Are the study results completely defined?
✓ Do the results help to solve the requirements traceability problems?
✓ What types of validity threats are defined for the results?

Analysis
✓ How the data has been analyzed?
✓ What type of analysis techniques are used e.g. SWOT analyses, risk analysis?
✓ If the article contains some framework then is it validated in some industrial setting?

Discussion / Conclusion
✓ Are negative findings properly reported?
✓ Is there any a limit or restrictions imposed on the conclusions claims?

2.1.2.1.7 Data Extraction Strategy
In order to obtain the information from each primary study following data extraction form will be used.

1. General Information about Research Article

1.1 Article Title
1.2 Author(s) Name(s)
1.3 Journal /Conference/Conference Proceedings
1.4 Search Terms Used to Retrieve Research Article
1.5 Retrieval Database of Research Article
1.6 Date of Publication

2. Specific Information about Research Article

2.1 Study Environment
2.1.1 Industrial
2.1.2 Academia

2.2 Research Methodology
2.2.1 Action Research
2.2.2 Experiment
2.2.3 Case Study
2.2.4 Survey

2.3 Subjects
2.3.1 Professionals
2.3.2 Students
2.3.3 Number of Subjects
2.3.4 Subject Selection

2.4 Requirements Traceability
2.4.1 Definition of Requirements Traceability
2.4.2 Challenges with Requirements Traceability
2.4.3 Problems with Requirements Traceability
2.4.4 Solutions of Requirements Traceability Problems
2.4.4.1 Model / Framework for Requirements Traceability
2.4.4.2 Requirements Traceability Tools
2.4.4.3 Number of Requirements Traceability Techniques Used in Model / Framework
2.4.4.4 Name(s) of Requirements Traceability Techniques Used in Model / Framework
2.4.4.5 Evidence Regarding Validation of Proposed Model / Framework

2.4.5 Requirements Traceability Techniques
2.4.5.1 Evaluation of Requirements Traceability Techniques
2.4.5.2 Comparison of Requirements Validation Techniques

2.5 Validity Threats
2.5.1 Conclusion Validity
2.5.2 Construct Validity
2.5.3 Internal Validity
2.5.4 External Validity

2.1.2.1.8 Synthesis of Extracted Data

In the systematic review, data synthesis is done by collecting and summarizing the results of the included primary studies. The studies included in systematic review are different from each other based on their methodology and outcomes. These types of studies are known as heterogeneous. Due to the heterogeneous nature of the data qualitative synthesis is proposed. The qualitative synthesis will be conducted by reading and analyzing the research articles. All the results will be mentioned in accordance with the research questions given in the review protocol. In order to obtain the information from each primary study the data extraction forms will be used.

2.1.3 Evaluating a Review Protocol

The review protocol is the critical element of the systematic review and therefore an agreed validation process should be carried out for evaluating the protocol. Kitchenham et.al. [16] recommends conducting pilot searches to identify the potential primary studies using search terms and search resources which are defined in the review protocol.

The review protocol was peer-reviewed by two researchers form Blekinge Institute of Technology, Sweden. These researchers have experience in conducting systematic literature reviews. The authors of the review protocol contacted the librarian at Blekinge Institute of
Technology for defining search terms, identifying search resources and conducting pilot searches.

2.2 **Conducting the Review**

The steps involved in conducting a systematic literature review are briefly explained below.

2.2.1 **Identification of Research**

The aim of a systematic review is to find as many primary studies related to the particular research questions as possible using an unbiased search strategy [16]. This unbiased search strategy is defined in the review protocol.

In order to define search terms, the research questions are divided into individual facets like population, intervention, outcomes and study designs. The synonyms of the search terms are identified. Search terms are also constructed using Boolean AND to join search terms and OR to include synonyms.

In software engineering population is used to describe following [16],

- A specific software engineering role e.g. requirements engineers, requirements managers.
- A category of software engineers e.g. novice requirements engineer.
- An application area e.g. requirements traceability, requirements engineering.
- An industry group e.g. IT companies, telecommunication operators.

An intervention is the software engineering methodology, procedure, technology or tool that addresses a specific issue [16]. For example requirements traceability techniques is an intervention supporting traceability.

Comparison is the software engineering methodology, tool or technique which is used to compare with intervention [16]. For example, comparison between requirements traceability practices in two telecommunication companies.

Outcomes are the factors of interest for practitioners or researchers [16]. For example improved traceability, returns on investment by using requirements traceability can be of interest to traceability researchers.

The trial searches were conducted in consultation with the librarian at Blekinge Institute of Technology using search terms defined in the review protocol. Different electronic search resources were selected as no single resource returns all the primary studies related to the area under study.

The systematic reviews are subject to publication bias. Publication bias refers to the phenomenon that positive results are more likely to be published than negative results [16]. In addition to the online search, proceedings and journals were manually scanned to eliminate publication bias. These proceeding and journals are specified in the review protocol under search strategy.

The Endnote was used to manage the large number of references of primary studies obtained by the systematic review. Each article was downloaded after inserting its details in the Endnote. This also eliminated the possibility of double insertion of a particular article.

2.2.2 **Selection of Primary Studies**

The selection of primary studies is a two-stage process. In the first stage the title, abstract and conclusion of a paper is studied and irrelevant papers are rejected. Then in the second stage the selected research papers are reviewed based on the inclusion/ exclusion criteria defined in the review protocol to obtain a final list of primary studies.

In the systematic literature more than 3087 articles were scanned and initially 72 articles were selected. Then after applying inclusion/exclusion criteria 51 articles were finally selected for review. These articles are given Table 2.
Table 2: Articles selected for systematic review

<table>
<thead>
<tr>
<th>No</th>
<th>Ref#</th>
<th>Article</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[4]</td>
<td>Requirements Traceability – When and how does it Deliver more than it Costs?</td>
</tr>
<tr>
<td>2</td>
<td>[5]</td>
<td>A Heterogeneous Solution for Improving the Return on Investment of Requirements Traceability</td>
</tr>
<tr>
<td>3</td>
<td>[6]</td>
<td>Just Enough Requirements Traceability</td>
</tr>
<tr>
<td>5</td>
<td>[8]</td>
<td>An Evaluation of Traceability Approaches to Support Software Evolution</td>
</tr>
<tr>
<td>6</td>
<td>[11]</td>
<td>Extended Requirements Traceability: Results of an Industrial Case Study</td>
</tr>
<tr>
<td>7</td>
<td>[21]</td>
<td>A Case Study on Value-based Requirements Tracing</td>
</tr>
<tr>
<td>8</td>
<td>[22]</td>
<td>Factors Influencing Requirements Traceability Practice</td>
</tr>
<tr>
<td>9</td>
<td>[23]</td>
<td>Requirements Traceability in Automated Test Generation Application to Smart Card Software Validation</td>
</tr>
<tr>
<td>11</td>
<td>[25]</td>
<td>Toward Improved Traceability of Non-Functional Requirements</td>
</tr>
<tr>
<td>12</td>
<td>[26]</td>
<td>Requirements Tracing</td>
</tr>
<tr>
<td>13</td>
<td>[27]</td>
<td>Requirements Traceability to Support Evolution of Access Control</td>
</tr>
<tr>
<td>14</td>
<td>[28]</td>
<td>A Feature-Oriented Requirements Tracing Method: A Study of Cost-benefit Analysis</td>
</tr>
<tr>
<td>15</td>
<td>[29]</td>
<td>Experiences Using Scenarios to Enhance Traceability</td>
</tr>
<tr>
<td>16</td>
<td>[30]</td>
<td>Pre-Requirement Specification Traceability: Bridging the Complexity Gap through Capabilities</td>
</tr>
<tr>
<td>17</td>
<td>[31]</td>
<td>Rule-based Generation of Requirements Traceability Relations</td>
</tr>
<tr>
<td>18</td>
<td>[32]</td>
<td>Tracing Cross-Cutting Requirements via Context-Based Constraints</td>
</tr>
<tr>
<td>19</td>
<td>[33]</td>
<td>Using Scenarios to Support Traceability</td>
</tr>
<tr>
<td>20</td>
<td>[34]</td>
<td>Enhancing Traceability Using Ontologies</td>
</tr>
<tr>
<td>21</td>
<td>[35]</td>
<td>Reconstructing Requirements Coverage Views from Design and Test using Traceability Recovery via LSI</td>
</tr>
<tr>
<td>23</td>
<td>[37]</td>
<td>Recovery of traceability links between software documentation and source code</td>
</tr>
<tr>
<td>24</td>
<td>[38]</td>
<td>Automating performance-related impact analysis through event based traceability</td>
</tr>
<tr>
<td>26</td>
<td>[40]</td>
<td>Tracing requirements to defect reports: an application of information retrieval techniques</td>
</tr>
<tr>
<td>27</td>
<td>[41]</td>
<td>REquirements TRacing On target (RETRO): improving software maintenance through traceability recovery</td>
</tr>
<tr>
<td>28</td>
<td>[42]</td>
<td>Scenario advisor tool for requirements engineering</td>
</tr>
<tr>
<td>29</td>
<td>[43]</td>
<td>XTraQue: traceability for product line systems</td>
</tr>
<tr>
<td>31</td>
<td>[45]</td>
<td>A Reusable Traceability Framework using Patterns</td>
</tr>
<tr>
<td>32</td>
<td>[46]</td>
<td>Traceability for System Families</td>
</tr>
<tr>
<td>33</td>
<td>[47]</td>
<td>Tool support for computer-aided requirement traceability in architectural design: The case of DesignTrack</td>
</tr>
<tr>
<td>34</td>
<td>[48]</td>
<td>Automating Requirements Traceability: Beyond the Record &amp; Replay Paradigm</td>
</tr>
<tr>
<td>36</td>
<td>[50]</td>
<td>Design Traceability</td>
</tr>
<tr>
<td>37</td>
<td>[51]</td>
<td>Overcoming the Traceability Benefit Problem</td>
</tr>
<tr>
<td>38</td>
<td>[52]</td>
<td>Modeling Functional Requirements to Support Traceability Analysis</td>
</tr>
</tbody>
</table>
The rejected articles are included in Appendix 1. The search was carried out in two steps. During the first step the selected electronic resources were explored online using the search terms defined in the review protocol. The articles related to requirements traceability were downloaded and their details were entered into Endnote. In the second step the selected conference proceedings and journals were searched manually year by year. The articles related to requirements traceability were first searched in Endnote. If they were not found in Endnote they were downloaded. This was done to ensure that no key article is missed during the systematic review.

Table 3 displays the statistical data of research papers related to traceability in the selected conference proceedings and journals mentioned in Table 1. These articles were already downloaded during the online search and their details were present in Endnote. So there was no duplication.

<table>
<thead>
<tr>
<th>Sr No</th>
<th>Journal</th>
<th>Total articles</th>
<th>Articles on R.T</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IEEE Transactions on Software Engineering (TSE)</td>
<td>751</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>ACM Transactions on Software Engineering Methodology (TOSEM)</td>
<td>139</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Springer Requirements Engineering Journal</td>
<td>144</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>Springer Innovations in Systems and Software Engineering</td>
<td>48</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Springer Software and Systems Modeling</td>
<td>106</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Springer Annals of Software Engineering</td>
<td>150</td>
<td>2</td>
</tr>
</tbody>
</table>

**Conference Proceedings**

<table>
<thead>
<tr>
<th>Sr No</th>
<th>Conference</th>
<th>Total articles</th>
<th>Articles on R.T</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ACM International Conference on Software Engineering (ICSE)</td>
<td>1272</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>IEEE International Conference on Requirements Engineering</td>
<td>357</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>IEEE International Symposium on Requirements Engineering</td>
<td>120</td>
<td>2</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>3087</strong></td>
<td><strong>29</strong></td>
</tr>
</tbody>
</table>

There were only three articles [41, 42, 43] that were found by manually exploring the journals mentioned in Table 1. These articles were downloaded and their details were added into the Endnote. This ensures the validity of systematic review as maximum effort was made not to miss any article related to requirements traceability. In systematic review total number of
articles related to requirements traceability was 72. The 29 articles mentioned in Table 3 are also included in the total article count.

Figure 5 represents the yearly distribution of research articles on requirements traceability during the period 1997 – 2007.

![Articles on RT](image)

**Figure 5: Year-wise publication of articles on requirements traceability during 1997-2007**

It is evident from Figure 5 that the number of publications on requirements traceability started to rise gradually after year 2000 reaching maximum level in year 2005. One of the reasons for this increase in research related to traceability is the international workshop on Traceability in Emerging Forms of Software Engineering (TEFSE) held in 2002 (TEFSE ‘0202), 2003 (TEFSE ‘03) and 2005 (TEFSE ‘05). In this workshop only research papers related to traceability were presented and later published electronically.

2.2.3 Study Quality Assessment

The study quality assessment is carried out to support inclusion / exclusion criteria. The authors developed a checklist for study quality assessment by dividing a research paper in introduction, method, results, analysis and discussion (IMRAD) as defined in section 2.1.2.6.

2.2.4 Data Extraction & Monitoring

The main objective of this phase is to design data extraction forms to accurately record the information obtained from primary studies [16]. Data extraction forms should be defined and piloted when the review protocol is defined, to reduce the opportunity for bias. Therefore the authors developed the data extraction with the review protocol.

During this phase one author extracted the data and the other author checked the extracted data. This strategy ensures that all important information is extracted from the research articles.

2.2.5 Data Synthesis

Data synthesis involves collecting and summarizing the results of the included primary studies [16]. The extracted data is synthesized to answer the research questions proposed in the review protocol. Data synthesis can be descriptive synthesis, quantitative synthesis and qualitative synthesis [16].
In descriptive synthesis extracted information should be tabulated in a manner consistent with the research questions defined in the review protocol. Tables should be structured to highlight the similarities or differences between the studies outcome [16]. If the results from studies are consistent with one another they are homogeneous and if the study results are inconsistent they are heterogeneous. The heterogeneous results should be tabulated to identify the sources of heterogeneity like sample size, study quality and study type.

Quantitative synthesis involves presenting the results in tabular form based on the sample size for intervention, difference between the mean values for each intervention and units used for measuring the effect [16].

Qualitative synthesis involves trying to integrate the studies comprising language results and conclusions, where different researchers have used terms and concepts with different meanings [16]. Kitchenham et al. describes three approaches for qualitative data synthesis.

1. **Reciprocal translation**
   Reciprocal translation is used when studies are about similar things and researchers are attempting to provide and additive summary [16].

2. **Refutational synthesis**
   Refutational synthesis is useful when studies are implicitly or explicitly refutations of each other [16]. In this type of qualitative synthesis individual studies and refutations are translated and analyzed.

3. **Line of argument synthesis**
   This type of qualitative synthesis is useful when researchers are concerned about what they can infer about from a topic as a whole from set of studies that focus on the part of the issue [16]. Line of argument synthesis is a two step method. In the first step individual studies are analyzed, and then an attempt is made in the second step to analyze the set of individual studies as a whole.

   The nature of software engineering systematic reviews seems to be qualitative [16]. Therefore authors selected line of argument synthesis as qualitative data synthesis approach for the systematic review.

### 2.3 Reporting the Review

Reporting the systematic review is a single phase stage. The results of the systematic review are reported in this phase according to the research questions defined in the review protocol. During the systematic review the relevant data is gathered using data extraction forms. The finally this data is reported and synthesized using appropriate data synthesis approaches.
3 SYSTEMATIC REVIEW RESULTS

The results of the systematic review are presented below according to our research questions.

3.1 Definitions of Requirements Traceability

Gotel & Finkelstein defined requirements traceability as: "Requirements traceability refers to the ability to describe and follow the life of a requirement, in both a forward and backward direction (i.e., from its origin, through its development and specification, to its subsequent deployment and use, and through periods of on-going refinement and iteration in any of these phases)" [6, 7, 8, 11, 21, 23, 24, 26, 27, 29, 30, 32, 39, 48, 49, 50, 53, 63]. This definition seems to be more precise and comprehensive definition as compared to other definitions reported in the literature. That’s why it is referred by various researchers. It describes both pre-RS traceability (It refers to all aspects of requirements before documenting them in requirements specification) and post-RS traceability (It refers to all aspects of requirements after documenting them in requirements specification).

Ramesh & Jarke defined requirements traceability as: “the ability to relate requirements specifications with other artefacts created in the development life-cycle of a software system” [31]. This definition deals with the traceability from requirements to upcoming software artifacts but it ignores pre-RS traceability.

The ANSI/IEEE standard 830-1998 defines traceability as: “a software requirements specification is traceable if (i) the origin of each of its requirements is clear and if (ii) it facilitates the referencing of each requirement in future development or enhancement documentation” [38, 39, 52, 63]. This definition covers both pre-RS traceability and post-RS traceability.

The US department of defense standard DoD Std-2167A defines traceability as: “Traceability means that the document in question is in agreement with a predecessor document to which it has a hierarchical relationship. Traceability has five elements.

- the document in question contains or implements all applicable stipulations of the predecessor document,
- A given term, acronym, or abbreviation means the same thing in all documents,
- A given item or concept is referred to by the same name or description in the documents,
- All material in the successor document has its basis in the predecessor document, that is, no untraceable material has been introduced, and
- The two documents do not contradict one another” [52].

This definition provides a solid base to the researchers to understand and interpret requirements traceability. This definition covers almost all aspects of requirements traceability.

Ramesh and Jarke [7] present the definitions given by different researchers like Edwards & Howell, Palmer, and Hamilton & Beeby. Edwards & Howell define requirements traceability as: “a technique used to provide a relationship between the requirements, the design and the final implementation of the system” [7]. This definition reflects the traces from requirement to the other artifacts which mean that it focus on post-RS traceability but on the other hand it ignores the pre-RS traceability.

Palmer defined traceability as: “traceability gives essential assistance in understanding the relationship that exists within and across software requirements, design and implementation”. This definition is similar to the above definition by Edwards and Howell. It focuses only on the post-RS traceability.

Hamilton & Beeby define traceability as: “the ability to discover the history of every feature of a system so that the changes in the requirements can be identified”. This definition covers both pre-RS traceability and post-RS traceability. This is due to the fact that both these aspects of requirements traceability identify change in requirements.
Bashir and Qadir [63], report the definitions of requirements traceability by researchers like Gotel & Finkelstein, Ramesh, Spanoudakis and Murray. Spanoudakis defines traceability as: "the ability to relate requirements specifications with other artifact created in the development life-cycle of a software system". This definition refers only post-RS traceability and ignores pre-RS traceability.

Murray defines traceability as: "the ability to identify requirements at different levels of abstraction and to show that they have been implemented and tested". This definition only focuses on post-RS traceability.

Ramesh define traceability as: "property of a system description technique that allows changes in one of the three system descriptions - requirements, specifications, and implementation - to be traced to the corresponding portions of the other descriptions". This definition only focuses on the post-RS traceability.

Table 4: Articles defining requirements traceability

<table>
<thead>
<tr>
<th>No.</th>
<th>Ref. No.</th>
<th>Article Name</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[21]</td>
<td>A Case Study on Value-based Requirements Tracing</td>
<td>Gotel &amp; Finkelstein</td>
</tr>
<tr>
<td>2</td>
<td>[23]</td>
<td>Requirements Traceability in Automated Test Generation Application to Smart Card Software Validation</td>
<td>Gotel &amp; Finkelstein</td>
</tr>
<tr>
<td>4</td>
<td>[26]</td>
<td>Requirements Tracing</td>
<td>Gotel &amp; Finkelstein</td>
</tr>
<tr>
<td>5</td>
<td>[27]</td>
<td>Requirements Traceability to Support Evolution of Access Control</td>
<td>Gotel &amp; Finkelstein</td>
</tr>
<tr>
<td>6</td>
<td>[29]</td>
<td>Experiences Using Scenarios to Enhance Traceability</td>
<td>Gotel &amp; Finkelstein</td>
</tr>
<tr>
<td>7</td>
<td>[30]</td>
<td>Pre-Requirement Specification Traceability: Bridging the Complexity Gap through Capabilities</td>
<td>Gotel &amp; Finkelstein</td>
</tr>
<tr>
<td>8</td>
<td>[31]</td>
<td>Rule-based Generation of Requirements Traceability Relations</td>
<td>Ramesh &amp; Jarke</td>
</tr>
<tr>
<td>9</td>
<td>[32]</td>
<td>Tracing Cross-Cutting Requirements via Context-Based Constraints</td>
<td>Gotel &amp; Finkelstein</td>
</tr>
<tr>
<td>11</td>
<td>[38]</td>
<td>Automating performance-related impact analysis through event based traceability</td>
<td>IEEE std</td>
</tr>
<tr>
<td>12</td>
<td>[48]</td>
<td>Automating Requirements Traceability: Beyond the Record &amp; Replay Paradigm</td>
<td>Gotel &amp; Finkelstein</td>
</tr>
<tr>
<td>14</td>
<td>[50]</td>
<td>Design Traceability</td>
<td>Gotel &amp; Finkelstein</td>
</tr>
<tr>
<td>16</td>
<td>[51]</td>
<td>Overcoming the Traceability Benefit Problem</td>
<td>Paul &amp; Steve</td>
</tr>
<tr>
<td>17</td>
<td>[52]</td>
<td>Modeling Functional Requirements to Support Traceability Analysis</td>
<td>DOD + IEEE std</td>
</tr>
<tr>
<td>21</td>
<td>[53]</td>
<td>Supporting Event Based Traceability through High-Level Recognition of Change Events</td>
<td>Gotel &amp; Finkelstein</td>
</tr>
</tbody>
</table>
The Table 4 presents the statistical data of the number of articles defining traceability. The results show that 19 articles refer to the definition of traceability given by Gotel & Finkelstein.

3.1.1 Analysis

In requirements traceability requirements are evidently connected to their origin and to other software artefacts produced during software development life cycle. On the basis of systematic review (Section 3.1) we found that the definition of requirements traceability provided by Gotel & Finkelstein, Hamiltion & Beeby, DOD and ANSI/IEEE standard 830-1998 are comprehensive and state-of-art definitions.

The articles containing definitions of requirements traceability gathered by the systematic review are presented in Table 4. In order to perform analysis of requirements traceability definitions we add two parameters “scope of definition” and “coverage” as columns in Table 4. The column “scope of definition” provides the aspect of traceability i.e. pre-RS traceability or post-RS traceability. The column “coverage” can have two values, complete and partial. A coverage is complete if the definition deals with both the aspects of traceability where as coverage is partial if the definition deals with only one aspect of traceability.

Table 5: Overview of Requirements Traceability Definitions

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Ref. No.</th>
<th>Definition</th>
<th>Scope of Definition</th>
<th>Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[6, 7, 8, 11, 21, 23, 24, 26, 27, 29, 30, 32, 39, 48, 49, 50, 51, 53, 63]</td>
<td>Gotel &amp; Finkelstein</td>
<td>Pre-RS traceability and Post-RS traceability</td>
<td>complete</td>
</tr>
<tr>
<td>2</td>
<td>[31]</td>
<td>Ramesh &amp; Jarke</td>
<td>Post-RS Traceability</td>
<td>partial</td>
</tr>
<tr>
<td>3</td>
<td>[38, 39, 52, 63]</td>
<td>ANSI / IEEE std 830-1998</td>
<td>Pre-RS traceability and Post-RS traceability</td>
<td>complete</td>
</tr>
<tr>
<td>4</td>
<td>[52]</td>
<td>DOD</td>
<td>Pre-RS traceability and Post-RS traceability</td>
<td>complete</td>
</tr>
<tr>
<td>5</td>
<td>[7]</td>
<td>Hamilton &amp; Beeby</td>
<td>Pre-RS traceability and Post-RS traceability</td>
<td>complete</td>
</tr>
<tr>
<td>7</td>
<td>[63]</td>
<td>Spanoudakis</td>
<td>Post-RS Traceability</td>
<td>partial</td>
</tr>
<tr>
<td>8</td>
<td>[63]</td>
<td>Murray</td>
<td>Post-RS Traceability</td>
<td>partial</td>
</tr>
<tr>
<td>9</td>
<td>[63]</td>
<td>Ramesh</td>
<td>Post-RS Traceability</td>
<td>partial</td>
</tr>
</tbody>
</table>

Hence the conclusion drawn from Table 4 is that requirements traceability is fully implemented when it both aspects of pre-RS traceability and post-RS traceability are covered. Secondly the definition of Gotel & Finkelstein, Hamiltion & Beeby, DOD and ANSI/IEEE standard 830-1998 completely defines traceability.

The pie chart in Figure 6 represents the statistics related to the definition of requirements traceability providing complete coverage. It is evident form Figure 6 that Gotel & Finkelstein’s definition of requirements traceability is dominant in more than 80% of the research reported in the systematic review.

Although the definition of Gotel & Finkelstein is complete and widely referenced but it does not answers the questions like how much traceability is enough? What is the right kind of traceability for a particular project?. It is interesting to know that these questions are still unanswered by the current research on requirements traceability.
We do not aim to make our own definition of requirements traceability by these collected definitions. The idea behind this entire definition logging is to get clear understand of traceability based on research and standards like DOD and IEEE std-830-1998.

Hence it is important to know that in order to implement traceability properly its both aspects of pre-RS and post-RS traceability should not be ignored. Our proposed framework of requirements traceability provides support for both these aspects of requirements traceability.

3.2 Challenges/Issues in Implementation of Requirements Traceability

In [6, 62] Cleland et al. discuss the results of a survey conducted by Gotel and Finkelstein in 1994. Various traceability problems were identified in this survey such as informal development methods, insufficient resources, time and cost for traceability, lack of coordination between people which were responsible for different traceable artefacts, lack of training for traceability, imbalance between benefits obtained & efforts spent for implementing traceability practices, failure to follow standards. Cleland et al. [62] comments that all of these issues are intensified by the challenges of todays distributed development environment. In order to solve some of these issues Cleland et al. [62] proposed a traceability technique named ‘Event-based Traceability (EBT)’. This technique creates the links between software artefacts after change request. EBT alleviate the coordination efforts required for the maintenance of software artefacts. Cleland [6] also recommends Information Retrieval (IR) methods in automated requirements traceability.

In another article [4] Cleland presents the challenges in requirement traceability from different researcher’s perspectives as: “What is the right kind of cost effective traceability for this project? ” and “How much traceability is enough?”. Cleland [4], present issues reported by various researchers like: Andrea Zisman, Jane Huffman and Jerme Dick. Andrea Zisman states that “There seem to be no arguments concerning the potential value that traceability can add to the development of software systems. However, traceability is rarely utilized as its cost is perceived to exceed its benefits”. She comments that “high cost of traceability results from the difficulty of automatically generating traceability relations with clear and precise semantics; the heterogeneity and large number of artefacts that are created during the development of software systems; and the lack of correctness and completeness of traceability relations”. Jane Huffman
states that "even with the advent of the ‘perfect’ tracing tool capable of generating traceability between artefact levels ‘on demand’ at the press of a button”. But the main question that how we can implement requirement traceability in effective manners is still unanswered. She recommends that “these answers must come from the stakeholders” and that we need “a way for stakeholders to approach such questions.”

In article [59] Gotel & Morris reports traceability problems like: requirements change by user, and availability of less contextual information in decision making about requirements. Gotel & Morris suggests a new dimension in the field of requirement traceability and media to solve these problems. They purpose a theoretical framework which helps to select the appropriate media for recording requirement-related information.

In article [51] Paul & Steve present some problems which were faced in execution of requirements traceability. According to them various project managers and team members perceive that requirements traceability does not offer immediate benefit to development process. Therefore requirement traceability is kept at low priority. They proposed a new method named as ‘Traceable Development Contract (TDC)’ to overcome traceability benefit problems. TDC is used to control interaction of development teams and to document traceability relationships.

In article [60] Cleland et al. identified manual construction and maintenance of traceability matrix proved to be costly for various industrial practices. Therefore, it is common perception that traceability is not feasible from financial point of view. In order to solve this problem, dynamic retrieval methods are used to automate the generation of traceability links. These methods are based on ‘Information Retrieval (IR)’ methods.

In article [5] Cleland et al. present some challenges reported by Gotel & Finkelstein. Gotel & Finkelstein report that, lack of coordination between team members results into failure of maintaining links between artefacts. Most of the time developers think that traceability costs more than it delivers. According to Domges et al. excessive usage of traceability can lead to confusion of the links. More links are not easy to maintain and use. Cleland et al. [5] present a combination of traceability techniques in a framework named ‘Traceability for Complex Systems (TraCS)’. They argue that TraCS helps to implement requirement traceability practices in cost effective manner and bring significant value to an organization.

Gotel & Finkelstein [11] reports the results of an empirical study related to the problems of requirements traceability. They identify set of problematic questions such as: who identify or discover a requirement and how?, who was responsible for the requirement in start and who is responsible currently?, who will take care for change(s) in requirements?, and what will be the effect on project in terms of knowledge loss, if some individual or the whole group leave the company?. Gotel & Finkelstein propose a model using contribution structures to solve these issues. Contribution structures reflect the network of people who participate in production of artefacts in requirement engineering. This model extends the artefact-based traceability with traceability of people involved in requirements engineering. This model is successfully implemented in communication-related company as a case study [11]. Results of the proposed model were encouraging. The employees of the company comment that the proposed model identifies the relevant people to rectify the specific problems regarding changes in requirements. It successfully guides towards handling staff turnover issue. The main contribution of the model is that it provides information about social roles and social relations.

In article [58] Bjarne presents the experience gained from a project related to the improvement in requirement management, specifically in requirement traceability. This project was carried out in Alcatel Telecom, Norway. Initially the problems related to requirement management were identified as information transfer from contract to specification phase as well as the impact analysis of requirement change was not satisfactory or adequate. In this regard three requirements management tools DOORS, RTM and Requisite Pro were evaluated. Finally DOORS was recommended to solve traceability related issues.

In article [22] Ramesh identified environmental, organizational and technical factors influencing implementation of requirements traceability. The environmental factors include inability to use available technologies for requirements traceability such as reluctance to construct manual Requirements Traceability Matrix (RTM) by the employees. Organizational factors include compliance with standards strictly demanding requirements traceability like
CMMI level 3. Technical factors include ad-hoc practices and staff employed in organizations. Ramesh [22] proposes tools support, training, and change in system development policies to overcome these issues. Change in system development policies includes willingness of management to implement requirements traceability.

In article [21] Heindl & Biffil identify cost associated with traceability as a factor which obstruct its implementation. The traceability cost gets significantly increased if company uses requirements traceability tools. The reason for this increase in cost is that existing traceability techniques do not differentiate between high value and low value requirements. The high value requirements are those which are more important as compared to other requirements. They propose a method called ‘Value Based Requirements Traceability (VBRT)’ to solve this cost related issue of requirements traceability. The VBRT approach identify high value requirements based on parameters like number of artifacts, number of traces (links between software artifacts), and number of requirements. They have successfully implemented this approach in an industrial case study.

In article [24, 25] Cleland et al. identify that organizations fail to trace Non-Functional Requirements (NFR) like performance, security and usability. This is due to the fact that NFR have a global impact on the software system and extensive network of interdependencies and trade-offs exist between them. In order to overcome these NFR related traceability issues an approach called ‘Goal Centric Traceability (GCT)’ is proposed. An industry-based experiment was conducted to verify this technique. The experimental results reveal that this approach is successful for managing traceability in NFR.

In article [30] Ravichandar et al. identify one major problem of tracing requirements back to their sources. The system validation testing is performed against requirements therefore; there should be a technique to trace requirements back to their sources. Pre-RS traceability facilitates tracing requirements to their origins. Less work has been done in this area. Ravichandar et al. [30] propose a technique for pre-RS traceability. This approach facilitates tracing requirements back to their sources or origins.

Table 6: Articles reporting issues related to requirements traceability

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Ref. No</th>
<th>Issues</th>
<th>Proposed Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[6, 62]</td>
<td>Informal development methods, Insufficient resources, Time and cost for traceability, Lack of coordination between people, Failure to follow standards.</td>
<td>Event-Based Traceability (EBT) technique, Information Retrieval (IR) methods.</td>
</tr>
<tr>
<td>2</td>
<td>[4]</td>
<td>How much traceability is enough?, What is the right kind of traceability for a particular project?</td>
<td>Still unanswered.</td>
</tr>
<tr>
<td>3</td>
<td>[59]</td>
<td>Requirements change by user, Less appropriate information is available for making decision with requirements.</td>
<td>Framework using media to record requirements related information and traceability links.</td>
</tr>
<tr>
<td>4</td>
<td>[51]</td>
<td>Requirement traceability does not offer immediate benefit to development process.</td>
<td>A new method named Traceable Development Contract (TDC) is proposed.</td>
</tr>
<tr>
<td>6</td>
<td>[5]</td>
<td>Lack of coordination between team members, Developers think that traceability costs more then it delivers, Excessive use of traceability generate more links which are not easy to manage.</td>
<td>A framework named Traceability for Complex Systems (TraCS) is proposed for implementation of traceability in cost-effective manner.</td>
</tr>
</tbody>
</table>
Some problematic questions are identified as issues like who identify a requirement and how?, who was responsible for the requirement in start and who is responsible currently, who will take care for change(s) in requirements, what will be the effect on project in terms of knowledge loss, if some individual or the whole group leave from company.

A model was proposed by Gotel & Finkelstein using contribution structures. This model reflects the network of people participating in requirements engineering activities including traceability.

Requirements management tools like DOORS, Requisite Pro, RTM.

Some recommendations were given such as tool support for traceability, training and change in system development policies.

Heindl & Biffil propose technique of VBRT to overcome cost related issues.

An approach called Goal Centric Traceability (GCT) is proposed to overcome these issues.

Ravichandar et al. propose a technique for pre-RS traceability.

3.2.1 Analysis

With the passage of time various researchers’ report different issues regarding implementation of requirements traceability practices. Among these some issues can be inferred as common such as cost, time, effort, and team coordination. In order to take care of these issues, various solutions have been proposed over the time such as new traceability techniques, frameworks, models, and various automated traceability tools.

The issues related to requirements traceability and their solutions can be classified into three types on the basis of systematic review results.

1. Issues addressed by academia.
   This type includes those issues whose solutions are proposed in academia. These issues are reported in Table 7.

2. Issues addressed by industry.
   This type includes those issues whose solutions are verified in industry. In this case the solutions were proposed in academia but tested in an industrial environment. These issues are reported in Table 8.

3. Unsolved issues.
   These are the issues which are still unsolved in industry and academia. The systematic review results identify that research community is still working to solve these traceability issues. These issues are reported in Table 9.
### Table 7: Issues Related to Requirements Traceability Addressed by Academia

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Ref. No</th>
<th>Issues</th>
<th>Proposed Solution (not verified in industry)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[62]</td>
<td>Informal development methods, Insufficient resources, Time and cost for traceability, Lack of coordination between people, Failure to follow standards.</td>
<td>EBT.</td>
</tr>
<tr>
<td>2</td>
<td>[59]</td>
<td>Requirements change by user, Less appropriate information is available for making decision with requirements.</td>
<td>Media recording framework.</td>
</tr>
<tr>
<td>3</td>
<td>[51]</td>
<td>Requirement traceability does not offer immediate benefit to development process.</td>
<td>Traceable Development Contract (TDC).</td>
</tr>
<tr>
<td>5</td>
<td>[5]</td>
<td>Lack of coordination between team members, Developers think that traceability costs more then it delivers, Excessive use of traceability generate more links which are not easy to manage.</td>
<td>Traceability for Complex Systems (TraCS) Framework.</td>
</tr>
<tr>
<td>6</td>
<td>[30]</td>
<td>Problems associated with tracing back to their sources.</td>
<td>Pre-RS requirements traceability technique.</td>
</tr>
</tbody>
</table>

### Table 8: Issues Related to Requirements Traceability Addressed by Industry

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Ref. No</th>
<th>Issues</th>
<th>Proposed Solution (verified in industry)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>[11]</td>
<td>Some problematic questions are identified as issues like who identify a requirement and how?, who was responsible for the requirement in start and who is responsible currently, who will take care for change(s) in requirements, what will be the effect on project in terms of knowledge loss, if some individual or the whole group leave from company.</td>
<td>Framework of Contribution Structure.</td>
</tr>
<tr>
<td>3</td>
<td>[58]</td>
<td>Requirements management issues in industrial projects like inadequate impact analysis, lack of information transfer.</td>
<td>Requirements management tools like DOORS, Requisite Pro.</td>
</tr>
<tr>
<td>4</td>
<td>[22]</td>
<td>Organizational, environmental and technical factors.</td>
<td>Some recommendations were given</td>
</tr>
<tr>
<td>5</td>
<td>[21]</td>
<td>Cost related to requirements traceability.</td>
<td>Technique of VBRT using case study in Siemens Austria.</td>
</tr>
<tr>
<td>6</td>
<td>[25]</td>
<td>Failure to trace Non-Functional Requirements (NFR) like security, performance and usability.</td>
<td>Technique Goal Centric Traceability (GCT) with the help of experimental evaluation in “Ice Breaker System”.</td>
</tr>
</tbody>
</table>

### Table 9: Issues Related to Requirements Traceability Un-solved in Academia & Industry

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Ref. No</th>
<th>Issues</th>
<th>Industry View &amp; Academia View</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[4]</td>
<td>How much traceability is enough?, What is the right kind of traceability for a particular project?</td>
<td>Still unanswered.</td>
</tr>
</tbody>
</table>
We have observed that all the proposed solutions against the specific issues faced by a company might not be feasible for another company. Therefore, every company seems to be tackling its own issues. We believe that there are few things which should be taken into account by a company. There should be a complete understanding about traceability issues among all related people. Furthermore there seems to be a need for a generic framework/model and generic traceability methods/approaches which are easy to implement for large, medium and small organizations equally.

3.3 Requirements Traceability Tools

The requirements traceability tools that were identified in the systematic review are briefly described below.

3.3.1 RETRO

REquirements TRacing On-target (RETRO) is a requirements traceability tool facilitating the automatic generation of Requirements Traceability Matrix (RTM). RETRO uses information retrieval (IR) methods and Graphical User Interface (GUI) front-end [6, 40, 41, 56].

Hayes et al. [41] conducted a case study with thirty graduate-level students of requirements engineering course at the University of Kentucky. The subjects in this case study were divided into groups. One group was tracing requirements manually using Requirements Traceability Matrix (RTM) while the other group was using RETRO. Students who had previous experience of traceability were placed in the group tracing requirements manually. Both these groups had to trace twenty-two requirements to fifty-two design elements. The results of the case study revealed that the students using RETRO produced most accurate results.

3.3.2 Rational Requisite Pro

Rational RequisitePro is a requirements management tool developed by IBM. It provides support to save software requirements specification (SRS) document, link requirements to use-case diagrams, and test cases. When change to requirements occur Rational RequisitePro identifies the corresponding software artefacts that are affected. It also provides traceability support.

It is currently in use in industry whereas researchers [5, 49, 53, 55, 57, 61, 62] have identified it as a requirements management tool which also support traceability.

3.3.3 DOORS

DOORS is a requirements management tool developed by Telelogic. It is used to capture, link, trace, analyze, and manage changes to the requirements. For requirements traceability DOORS provides easy ways to create and traverse the links between requirements for example a link can be created by simply drag and drop operation. DOORS is also helpful in change management, it immediately flags the changes that could impact other requirements. In addition to all this DOORS also support dynamic report generation.

From 1997 to 2007 several researchers have reported DOORS as an automated requirement management tool, which is also quite helpful in requirement traceability. The efficiency of DOORS is seems to be proven as it is mentioned and referred by several researchers [5, 7, 49, 51, 53, 55, 57, 58, 61, 62].

3.3.4 DesignTrack

According to [47] DesignTrack is a prototype tool supporting requirements traceability. It provides traceability between requirements and architectural design DesignTrack facilitates the organizations by providing an integrated design environment (IDE) for requirement
modeling/specification. DesignTrack is also used as form exploration medium. An architect can use this tool to manage issues of design requirements and other design tasks.

The primary goals of DesignTrack system as reported by [47] are: (i) provide an IDE for designers to manage requirement information along with form exploration, (ii) facilitate the designers to reuse previous requirement information, (iii) permit the designers to track the changes as they go through requirement specification or form exploration tasks.

3.3.5 TRAM

TRAM stands for Tool for Requirements and Architectural Management. TRAM is a tool for managing system architecture, system requirements and traceability between them. This tool is based on an information model identifying the relationship between requirements engineering and architectural design.

According to [44] TRAM is used in case studies giving positive results. But these positive results are not explicitly discussed in [44].

3.3.6 Scenario Advisor Tool

In software engineering scenarios are used to model system functionality. Scenarios act as mediators between requirements engineers and software artefacts like Software Requirements Specification (SRS), design documents. The scenario advisor tool provides traceability support between scenario models and requirements. It also facilitates generating new scenarios and scenario variations [42].

In article [42] an empirical study is reported to determine the usefulness of scenario advisor tool in scenario based traceability. An experiment was conducted with two groups. One group was using scenario advisor tool while the other was not using tool. The subjects of the group without tool support were eight postgraduate students and two researchers. In the group using scenario advisor tool the subjects were nine postgraduate students and one researcher. The members of both groups had no previous experience in traceability using scenarios.

According to [42], the results of this experiment reflect that scenario advisor tool helps users to write scenarios without any domain knowledge. Therefore, users can write better scenarios and can trace requirements to scenario models using this tool. In debriefing interviews some users reported that the tool could be improved if it provides a scenario template or a step-by-step procedure to write good scenarios. This was included in improvement plan for scenario advisor tool.

3.3.7 Other Traceability Tools

The requirements traceability tools listed in Table 10 were not described by the selected articles for systematic review. Neither their working nor any empirical evidence related to them is reported by the selected literature.

<table>
<thead>
<tr>
<th>Sr.No</th>
<th>Ref.No</th>
<th>Requirements Management/Traceability Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[7,62]</td>
<td>System Level Automation Tool for Engineers (SLATE)</td>
</tr>
<tr>
<td>2</td>
<td>[62]</td>
<td>CRADLE</td>
</tr>
<tr>
<td>3</td>
<td>[7,55,57]</td>
<td>RDD-100</td>
</tr>
<tr>
<td>5</td>
<td>[7]</td>
<td>RTS</td>
</tr>
<tr>
<td>6</td>
<td>[7]</td>
<td>Rtrace</td>
</tr>
<tr>
<td>7</td>
<td>[7]</td>
<td>Teamwork/RQT</td>
</tr>
</tbody>
</table>
3.3.8 Analysis

Requirements traceability tools help to maintain traceability by automatically generating links between various software artefacts. DOORS and Rational Requisite Pro are widely used requirements management tools which also provide traceability support. Tools like DesignTrack and TRAM provide traceability between requirements and architecture. On the other hand Scenario advisor tool provides traceability support between requirements and scenarios. RETRO is purely a traceability tool facilitating automatic generation of RTM.

The results of the systematic review reveal that all the above traceability tools except DesignTrack have been evaluated empirically. Furthermore by detailed study a trend can be identified about usage of tools for traceability. It seems that requirement management tools are widely used instead of pure traceability tool. This is due the fact that requirement management tools also provide good traceability support. Table 11 provides the comparison of requirements traceability tools.

<table>
<thead>
<tr>
<th>Sr.No</th>
<th>Ref.No</th>
<th>Tool Name</th>
<th>Functionality</th>
<th>Nature</th>
<th>Empirical Evidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[6,40,41,56]</td>
<td>RETRO</td>
<td>Facilitate automatic generation of RTM.</td>
<td>Traceability tool.</td>
<td>Case study in article [41]</td>
</tr>
<tr>
<td>2</td>
<td>[5, 49, 53, 55, 57, 61, 62]</td>
<td>Rational Requisite Pro</td>
<td>Provides support to link requirements to use-case diagrams, test cases and save SRS in its database.</td>
<td>Requirements management tool providing traceability support.</td>
<td>Currently used in industry.</td>
</tr>
<tr>
<td>3</td>
<td>[5, 7, 49, 51, 53, 55, 57, 58, 61, 62]</td>
<td>DOORS</td>
<td>It is used to capture, link, trace, and manage changes to requirements.</td>
<td>Requirements Management tool. It supports traceability.</td>
<td>Currently used in industry.</td>
</tr>
<tr>
<td>4</td>
<td>[47]</td>
<td>Design Track</td>
<td>Traceability between requirements and architectural design.</td>
<td>It is a requirements management tool supporting requirements traceability.</td>
<td>No Empirical Evidence</td>
</tr>
<tr>
<td>5</td>
<td>[44]</td>
<td>TRAM</td>
<td>Managing system architecture and system requirements. It supports traceability between them.</td>
<td>It supports traceability.</td>
<td>Case studies</td>
</tr>
<tr>
<td>6</td>
<td>[42]</td>
<td>Scenario advisor tool</td>
<td>Traceability between scenario models and requirements.</td>
<td>Basically it helps in writing scenarios but It supports traceability.</td>
<td>Experiment</td>
</tr>
</tbody>
</table>

Based on the results of systematic review the requirements traceability tools can be classified into three categories.

1. Requirements management tools providing traceability support.
   This category includes those tools which are developed to facilitate requirements management activities. Traceability is a part of requirements management process therefore these tools also supports traceability.
   This category includes tools like Rational requisite pro, DOORS, Design track and Scenario advisor tool.
2. Requirements traceability tools.
   This category includes those tools which are developed purely for managing requirements traceability.
   This category includes tools like RETRO and TRAM.

3. Other requirements management/traceability tools.
   This category includes those tools which are only reported by the systematic review articles. The articles gathered by the systematic review does not describe their working or empirical evidence regarding their use.
   This category includes tools given in Table 10 (Section 3.3.7).

3.4 Requirements Traceability Techniques
   The requirements traceability techniques that were identified in the systematic review are briefly described below.

3.4.1 Value Based Requirements Traceability (VBRT)
   The existing requirements traceability approaches makes no difference between requirements that are very valuable to trace and requirements that are less valuable to trace. This increase the efforts related to requirements traceability and therefore requirements traceability seems more costly to be implemented in practice [21]. The tracing value depends on several parameters like stakeholder importance, risk or volatility of requirement and the necessary tracing cost [21]. The VBRT takes into consideration these parameters.

   Ramesh [22] identifies two types of traceability users’ namely low-end traceability users and high-end traceability users. The low-end traceability users capture traceability information uniformly for all requirements. Therefore they treat traceability as expensive. On the other hand high end traceability users recognize that all requirements are not equal with respect to their criticality or significance. Therefore, they maintain traceability for critical project requirements to keep cost under control and achieve traceability benefits. VBRT uses the same analogy of high-end traceability users.

   The goal of VBRT is to identify traces or traceability links based on prioritized requirements [21]. The identification of traces early in the project life cycle is easier than in the later stages. VBRT reduces the traceability efforts by the prioritized requirements. The VBRT process consists of five steps as given below.

1. Requirements definition
   During requirements definition the project manager or requirements engineer analyzes the software requirements specification to identify atomic requirements [21]. A unique identifier is assigned to each requirement by the requirements engineer. The result of requirements definition step is a set of requirements and their IDs.

2. Requirements prioritization
   During requirements prioritization phase all stakeholders assess the requirements based on three parameters which are: the value, risk and effort of each requirement [21]. The result of this phase is an ordered list of prioritized requirements based on three priority levels.

3. Packaging of requirements
   This step is optional and it allows a group of architects to identify the clusters of requirements [21]. These clusters of requirements help to develop and refine architecture from a given set of requirements.

4. Linking of artefacts
   During this step the project team identifies traceability links between requirements and artefacts [21]. Important requirements are traced in more detail than other requirements. These important requirements can be identified from the list of prioritized requirements based on three
levels developed in requirements prioritization step. Over all traceability plan is the by-product of this phase.

5. Evaluation
The project manager can use traces for various purpose such as to estimate the impact of change [21]. Figure 7 provides the overview of Value Based Requirements Traceability (VBRT) process.

![Figure 7: Overview of VBRT process [21]](image)

**Empirical Evidence**
Heindl and Biffil [21] conducted a casestudy using VBRT at Siemens Austria. The project “Public Transport on Demand” consisting of 46 requirements was selected to compare full tracing and VBRT.

The results of this casestudy are summarized below.

a) The results of the casestudy reveal that focusing on most important requirements reduces efforts as compared to tracing all requirement. The high efforts of requirements tracing is the factor due to which project teams do not implement traceability [21]. This high effort can be reduced using the approach of VBRT which takes 35% less efforts in comparision to full tracing [21].
b) It is easier to capture traceability information in earlier phases of software development life cycle as compared to capturing traceability in later stages. For example if no traceability information is maintained during the project. If a change request occurs then traceability information is desired, and it takes more efforts to maintain traceability links.

c) The prioritization step of VBRT identifies most valuable requirements that are needed to be traced in more detail than others. The prioritization step of VBRT helps to reduce the traceability efforts. All stakeholders normally participate in requirements prioritization.

3.4.2 Feature Oriented Requirements Tracing (FORT)

Certain artifacts like software requirements specification, design documents, source code and test cases are produced during software development. When a ‘change request (CR)’ occurs at any stage during software development life cycle it is hard to discover on the part of software engineers the software artifacts affected by the change request. It is also hard for software engineers to construct and manage traceability links. The approach of feature-oriented requirements traceability reduces the hardship of managing traceability links by identifying them through prioritized requirements and by considering cost and efforts [28].

In order to understand the concept of feature oriented requirements tracing some terms need to be introduced as given below.

Features

Features are the key characteristics of the product [28]. These features can be classified on the basis of capabilities, domain technologies, implementation technologies and operating environments [28].

Feature Modeling

It is the process of identifying features and then organizing them in a model called feature model [28]. The activities involved in the feature modeling are defined below in Table 12.

<table>
<thead>
<tr>
<th>Steps</th>
<th>Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Domain Planning</td>
<td>1.1 Selecting a domain</td>
</tr>
<tr>
<td></td>
<td>1.2 Clarifying the boundary</td>
</tr>
<tr>
<td></td>
<td>1.3 Organizing the domain analysis trams</td>
</tr>
<tr>
<td></td>
<td>1.4 Making a domain directory</td>
</tr>
<tr>
<td>2. Feature Identification</td>
<td>2.1 Analyzing terminologies</td>
</tr>
<tr>
<td></td>
<td>2.2 Identifying product categories</td>
</tr>
<tr>
<td></td>
<td>2.3 List features in each categories</td>
</tr>
<tr>
<td></td>
<td>2.4 List product and features</td>
</tr>
<tr>
<td>3. Feature Organization</td>
<td>3.1 Organize features into a feature diagram</td>
</tr>
<tr>
<td></td>
<td>3.2 Reduce complexity of feature diagram</td>
</tr>
<tr>
<td>4. Feature Refinement</td>
<td>4.1 Refine feature model</td>
</tr>
<tr>
<td></td>
<td>4.2 Refine an abstract model</td>
</tr>
</tbody>
</table>

Capabilities

Capabilities are the user visible characteristics that can be identified as operations, nonfunctional characteristics and distinct services [28].

Domain Technologies

Domain technologies represent the way of implementing service or operation [28].

Implementation Techniques
These are the generic functions or techniques that are used to implement domain functions, services, and operations [28].

**Operating Environments**

These are the environments in which the applications are in use [28].

**Relationships**

There are three types of relationships in feature modeling [28]. The ‘Composed-of relationship’ is used when there is a whole-part relationship between a feature and its sub-features. In ‘Generalization/specialization relationship’ features are the generalizations of sub-features. Where as ‘Implemented-by relationship’ is used when one feature is necessary to implement the other feature.

The Figure 8 shows the meta-model based on feature modeling of requirements.

![Figure 8: Feature Modeling [28]](image)

**The Process of Feature Oriented Requirements Tracing**

The feature oriented requirements tracing process consists of five phases.

1. **Requirements definition**

   This phase consists of three activities analyzing requirements specification, identifying atomic requirements and assigning identifier to each requirement [28]. The aim of requirements definition phase is to normalize user requirements to map them to various artefacts. In order to achieve this aim the requirements specification is analyzed to identify atomic requirements. Then a unique identifier is associated with each requirement. The result of this phase is a list of requirements with identifiers [28].

2. **Feature modelling**

   This phase consists of three activities identifying categories and features, organizing feature diagrams and assigning requirements related to features [28]. According to [28], feature diagrams are made by identifying the categories in the target system and features in each category. At this stage relationships between features are also taken into consideration. Finally all requirements are assigned to each feature. The result of this phase is a feature diagram and list of features. The Guidelines in Table 12 are used for feature modelling.
3. Feature prioritization
This phase consists of two activities i.e. estimating value of requirements and ordering list of feature [28]. In feature prioritization phase stakeholders estimate the requirements based on value, risk and effort of each requirement. Then the features are prioritized. Feature oriented requirements traceability provides a scale for feature prioritization as shown in the table.

Table 13: Priority levels and classification of artefacts [28]

<table>
<thead>
<tr>
<th>Level</th>
<th>Granularity</th>
<th>Classification of Artefacts</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Low</td>
<td>Components</td>
</tr>
<tr>
<td>2</td>
<td>Medium</td>
<td>Class</td>
</tr>
<tr>
<td>3</td>
<td>High</td>
<td>Methods</td>
</tr>
</tbody>
</table>

4. Requirements linking
This phase consists of three activities assigning artefacts to related feature, fractionating implementation elements by level and establishing requirements traceability links [28].

According to [28], in requirements linking requirements traceability links are generated and all artefacts are assigned to the related features. In requirements linking phase implementation items are fractionated by granularity levels and then traceability links are established. These traceability links are actually the relationships among requirements, features and artefacts. Similarly important requirements are traced in more detail than less important requirements. This phase results in a list of traceability links.

5. Traceability links evaluation
This phase consists of two activities i.e. usability traceability links in development process and refining traceability links [28]. During this phase traceability links are used for conflict analysis, change impact analysis and consistency checking in the development process. Based on this evaluation the traceability links can be modified.

![Figure 9: Overview of Feature Oriented Requirements Tracing [28]](image)

Empirical evidence of cost benefit analysis
Article [28] provides an empirical evidence related to Feature Oriented Requirements Tracing. The feature oriented requirements tracing was applied in a case study to a car rental
system containing 9 components, 49 classes and 152 methods. The results of this case study are given in the table below.

**Table 14: Results of case study on car rental system [28]**

<table>
<thead>
<tr>
<th>Granularity</th>
<th>No. of Links</th>
<th>Traceability</th>
<th>Efforts of Generating Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>9</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Medium</td>
<td>49</td>
<td></td>
<td>28</td>
</tr>
<tr>
<td>High</td>
<td>152</td>
<td></td>
<td>100</td>
</tr>
</tbody>
</table>

No. of traceability links = l*m*n
Where ‘l’ is the number of components, ‘m’ is the number of classes and ‘n’ is the number of methods.

Efforts of generating links = l[*m,[*n]]/TR * 100

The case study results reveal that this method reduces the efforts for generating traceability links to 24% to 72% [28]. Based on this statistics this method is useful for reducing traceability link generation efforts in complex systems.

**Comparison of Feature Oriented Requirements Tracing and VBRT**

The Feature Oriented Requirements Tracing provides variability information by feature modelling. This variability is helpful to estimate the impact of requirements change. This technique reduces the efforts to create traceability links by prioritizing the features. This technique provides a tight relationship between requirements and artefacts by the help of an intermediate catalyst. This is also shown in the figure 8.

A comparison is made between the value-based requirements tracing and feature oriented requirements tracing and is shown in the table below [28].

**Table 15: Comparison of VBRT and FORT**

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Value Based Requirements Traceability (VBRT)</th>
<th>Feature Oriented Requirements Traceability (FORT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Well-defined process</td>
<td>Partially supports</td>
<td>Supports</td>
</tr>
<tr>
<td>Value added tracing</td>
<td>Supports</td>
<td>Supports</td>
</tr>
<tr>
<td>Feature modeling</td>
<td>No support</td>
<td>Supports</td>
</tr>
<tr>
<td>Variability consideration</td>
<td>No support</td>
<td>Supports</td>
</tr>
<tr>
<td>Efforts reduction</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Tool support</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Empirical Evidence</td>
<td>Case study at Siemens Austria</td>
<td>Case Study on car rental system</td>
</tr>
</tbody>
</table>

3.4.3 Pre-RS Requirements Tracing

In pre-requirements specification tracing (pre-RS tracing) requirements are traced back to their source. These sources are the user needs which are basically unstructured information. Requirements engineers use interviews, questionnaires or prototyping to gather user needs during the process of requirements elicitation. These needs are documented as requirements in Software Requirements Specification (SRS). Traceability between requirements and their sources is one of the biggest challenges faced by the research community [30]. In system validation the system is validated against requirements and in case of test failure it becomes necessary to trace requirements back to their sources.

The approach of Pre-RS traceability defined in [30] is based on Capabilities Engineering. The capabilities engineering is a process for developing change-tolerant systems by using functional abstractions known as capabilities.
Problem space
The problem space represents the conceptual region which is associated with the problem domain [30]. There are two important entities in the problem space. These are needs and directives.

The needs represent the user’s view of the system. The needs specify what is desired of the system from user’s perspective, expressed in the language of the problem domain [30]. Directives are the detailed characteristics of the system or they can be classified as requirements with context information [30]. There are two purposes of directives in problem space. First is to capture domain information and second is to facilitate progress from problem space to transition space [30].

In the problem domain needs are decomposed into directives. Decomposition is an intuitive process of recursively partitioning a process until an atomic level is reached [30]. Decomposition is achieved with the help of functional decomposition (FD) graph. Functional decomposition (FD) graph as shown in Figure 11. It is a directed acyclic graph represented as G= (V, E) where ‘V’ is the vertex and ‘E’ is the edge. In FD graph root represents the overall mission of the system and leaves represents the directives. The internal nodes between root and leaves are the functional abstractions.

The functional decomposition (FD) graph provides traceability links between needs and directives. When need is changed then corresponding directives that are affected can be identified by the help of functional decomposition (FD) graph.

Figure 10: Capabilities Engineering Process [30]

The Capabilities Engineering process is based on three phases.
1. Problem space
2. Transition space
3. Solution space
Figure 11: Functional decomposition (FD) graph [30]

**Transition space**

According to [30], transition space is defined as collective aggregation of system view, capabilities and problem domain. The two main entities in transition space are initial capabilities and optimized capabilities. Formulation and optimization are two capabilities engineering activities in transition space.

The initial capabilities are the functional abstractions with high cohesion and low coupling whereas the optimized capabilities are the constraints of technology feasibility and implementation schedules [30].

The activity of formulation identifies initial capabilities from all possible abstractions present in the FD graph. These initial capabilities show high cohesion and low coupling. Cohesion represents the togetherness of elements within the entity and coupling is the interdependences between elements.

The aim of optimization is to identify the set which best accommodates the constraints of schedule and technology [30]. The initial capabilities are the input for optimization activity. These initial capabilities are optimized capabilities as shown in Figure 10.

The input to transition space is the FD graph. The directives are converted into initial capabilities by applying the capabilities engineering CE activity known as formulation. The activities of formulation and optimization provide traceability links between directives, Initial capabilities and optimized capabilities.

**Solution space**

The solution space represents the technical area relevant to the system being developed [30]. The important entity in the solution space is the requirement as shown in Figure 10. In solution space requirement represents user’s expectations form the system and is subject to quality constraints. These quality constraints are testability, verifiability, accuracy, and un-ambiguity [30].

The input to the solution space is the optimized set of capabilities with their directives and these optimized capabilities are transformed into requirements. In capabilities engineering based pre-requirements traceability only the directives associated with the capabilities chosen for development are transformed into requirements [30].

The relevance value can be used to determine critical requirements while transforming directives to requirements. The relevance value represents the importance of a directive in
achieving the objective of its parent node [30]. For instance directive with relevance value 1 is mission critical and therefore the requirements associated with this directive are also mission critical. This relevance value provides cost effective way to maintain traces of only critical requirements.

Table 16: Pre-RS traceability using capabilities engineering approach

<table>
<thead>
<tr>
<th>CE Phase</th>
<th>Entities</th>
<th>CE Activity</th>
<th>Input</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem space</td>
<td>Needs, Directives</td>
<td>Decomposition</td>
<td>User needs</td>
<td>User</td>
</tr>
<tr>
<td>Transition space</td>
<td>Initial Capabilities, Optimized Capabilities</td>
<td>Formulation, Optimization</td>
<td>FD graph</td>
<td>System</td>
</tr>
<tr>
<td>Solution space</td>
<td>Finalized Capabilities (Requirements)</td>
<td>Transformation</td>
<td>Optimized Capabilities</td>
<td>System</td>
</tr>
</tbody>
</table>

Table 16 provides the summary of the pre-RS traceability process using capabilities engineering approach. In article [30] the authors does not provide any empirical evidence regarding the success of this approach. But they claim that this approach is very useful in requirements traceability.

3.4.4 Event Based Traceability (EBT)

Event Based Traceability (EBT) approach was proposed by Cleland et al. [61, 62]. The main reason for the development of EBT approach was to provide accurate maintenance of traceability relationships. Cleland et al [61] define the traceability relationships as publisher-subscriber relationship. In this relationship dependent object i.e. artefacts have to subscribe to their respective requirements on which they are dependent. Whenever a requirement change occurs, an event message is published, which is then notified to all dependent objects.

According to [53], there are three main components which participate in the whole process, requirement manager, event server and subscriber manager. As shown in Figure 12 requirement manager is used to manage the requirements and to publish the event message to event server, whenever a change request occurs. Event server is responsible for three main activities. It handles subscriptions from dependent objects. It is responsible for listening event messages from requirement manager and publishing or forwarding event messages to the relevant subscribers. Subscriber manager is responsible for listening event server and for receiving and managing the event notifications.

EBT handles both functional requirements and non-functional requirements [5, 38]. EBT provides solution for the traceability update problem and it can be used in combination with requirements management tools like DOORS and Rational Requisite Pro [62].
3.4.5 Information Retrieval (IR)

Information Retrieval (IR) [24, 57, 60] approach is used to automate the generation of traceability links. Commonly used IR methods include, Vector Space Model (VSM) Probabilistic Models and Latent Semantic Indexing (LSI). IR methods are based on similarity comparison and probabilistic value of two artefacts.

According to [8] IR methods include three general steps: (i) pre-processing; (ii) analyzing, indexing, creating its representation and archiving; (iii) analyzing incoming artefacts using some ranking algorithms. Whenever a pair of artefacts reaches over a specific rank, it is considered as candidate links which must be reported to the analyst to make the final decision. The analyst differentiates between the true and false links.

IR methods significantly reduce the effort required for creating traceability links between artefacts, but on the other hand it still requires significant efforts by the analyst.

The IR methods like VSM and LSI are used in the development of requirements traceability tool RETRO [36, 55]. The research work on IR methods has focused on tracing functional requirements. Whereas Cleland et al. [25] used IR methods in Goal Centric Traceability (GCT) for tracing non-functional requirements.

3.4.6 Rule Based (RB) Approach

Rule-based (RB) approach was purposed by Spanoudakis et al. in [8, 31]. The basic purpose of the RB approach is to automatically generate traceability links using rules. There are two traceability rules, i.e. requirement-to-object-model traceability (RTOM) rule and inter-requrement traceability (IREQ) rule.

According to [31] these rules are used for three specific documents, such as requirement statement document (RSD), use case documents (UCD), and analysis object model (AOM). RSD and UCD are traced to an AOM by using RTOM rules. IREQ rules are used for tracing between RSD and UCD. In RB approach all of the documents and both of the rules are presented in XML-based format. RB approach consists of four steps, which are grammatical tagging of the artefacts, converting the tagged artefacts into XML representations, generating traceability relations between artefacts, and generating traceability relations between different parts of the artefacts.

3.4.7 Hyper-text Based Approach (HB)

According to [8], Hyper-based approach (HB) approach is proposed by Maletic et al. This approach not only supports the complex linking, but also supports the versioning of links.

According to [8] XML is the main tool used by HB approach. Therefore all the links and models are converted into XML. In HP approach there are two types of models i.e. anchor model and target model. Causal, non-causal, and navigational links are created between anchor model and target model.

3.4.8 Feature-Model Based Approach (FB)

According to [8], Feature Model Based (FB) approach is described by Riebisch and Pashov. In feature modeling requirements are described as an overview and models as the variability of the product line. A feature model consists of graphs along with nodes and edges, while nodes are the features and edges are the features relations. Each feature represents a property of the product from customer’s point of view. There are three types of features such as functional features, interface features, and parameter features. Feature relations can be classified into three categories such as: Hierarchical relations in this case most important feature is placed at high position in hierarchy. Refinement relations are used to define the relations of generalization, specialization, and aggregation. Exclude relation defines the constraints between variable features which can influence the structure of the decision of the product.
3.4.9 **Scenario-Based Approach (SB)**

According to [8, 25] scenarios are used to model system functionality and to generate functional test cases. Scenarios-based test cases create a mapping between requirements and other artefacts like design and code. The traceability is established by mapping scenarios with the design elements. Scenarios are created to trace only the interesting cases therefore they might not provide complete coverage. However, scenarios are frequently used by several architectural assessment methods like architectural trade-off assessment method (ATAM), and software architecture assessment method (SAAM).

3.4.10 **Process Centered Environments**

In [25], Cleland reports a traceability technique named as Process Centered Engineering Environment (PCEE). This technique was used by Pohl et al. for tracing non-functional requirements. A PCEE is composed of three domains which are modelling, enactment and performance. Processes and traceability tasks are defined in modelling domain. The software engineering process and related traceability tasks are controlled by enactment domain. These software engineering and traceability tasks are implemented in performance domain.

The process centered environments can be used to trace both functional and non-functional requirements [25]. Non-functional requirements can be traced by connecting non-functional requirements and architectural assessment methods (AAM) in the enactment domain [25].

3.4.11 **Design Patterns**

According to Cleland [25] Gross and Yu proposed the technique of design patterns for tracing non-functional requirements, Softgoal Interdependency Graph (SIG), and the system design. This idea was utilized by Cleland et al. in a model to show traceability links between Softgoal Interdependency Graph (SIG) and underlying object oriented design. This model is based on the application of pattern detection algorithms within a subset of high-level explicitly traced classes. This technique supports traceability for any non-functional requirement that can be implemented as design pattern.

3.4.12 **Traceability Matrices**

Traceability matrices are commonly used in industry to define relationships between requirements document and other type of artefacts [25]. The other artefacts include design modules, code modules and test cases. In traceability matrices the links are manually created between requirements and other artefacts. Traceability matrices suffer from scalability and maintenance problems [25].

3.4.13 **Keywords and Ontology**

According to Cleland [25], Cysneiro et.al describes a technique named as ‘keywords and ontology’ which is based on Language Extended Lexicon (LEL). This approach provides traceability support between UML diagrams and non-functional requirements modelled in the form of goal tree. Keyword representing both domain and non-functional requirements are embedded in goal tree and UML diagrams. This approach requires maintenance and systematic use of keywords through out the evolution of system. Therefore there is no need to maintain a central traceability matrix.

3.4.14 **Aspect Weaving**

Aspect oriented programming AOP establishes traceability between aspects (lower level NFR) and code. According to [25] in ‘Aspect Oriented Programming (AOP)’ suitable concerns are modelled as aspects. In these aspects dispersed functionality is encapsulated into a single entity. This single entity is woven into the code with the help of a special compiler based on the set of aspect weaving rules.
Concerns can be categorized as functional and non-functional. Non-functional concerns include high level NFR such as maintainability, performance and security [25]. Nonfunctional concerns are less concrete to be implemented as aspects.

Functional concerns are more concrete concerns because their behavior is easily definable and they can be expressed in terms of aspect weaving rules [25]. The functional concerns include requirements like logging, authentication.

3.4.15 Goal Centric Traceability (GCT)

The traceability of Non-Functional Requirements (NFR) is difficult. This is due to the fact that extensive interdependencies and trade-offs exist between them. In article [24], Cleland-Huang et al. has proposed a technique of Goal Centric Traceability (GCT) to trace NFR. This technique has been successfully evaluated in case study with a real world project.

In GCT Softgoal Interdependency Graph (SIG) is used to model non-functional requirements as goals and operationalizations [24]. The SIG is a framework which helps developers to model NFR during software development. In SIG goals are the non-functional requirements to be satisfied where as operationalizations are development techniques or design techniques or implementation techniques which help to satisfy NFR [14].

The goal centric traceability model has four distinct phases namely goal modelling, impact detection, goal analysis and decision making [24]. These phases are briefly explained below.

1. Goal Modelling

The goal modelling occurs during elicitation, specification and architectural design of the system [24]. The goal modelling is sub-divided into two phases construct SIG and maintains SIG. In the construct SIG sub-phase the non-functional goals are modelled as softgoals, which are decomposed into operationalizations and then negotiated and agreed with the stakeholders [24].

The maintain SIG sub-phase maintains the SIG throughout the life of software system to accommodate the impact of change [24].

2. Impact Detection

According to [24] traceability links are established as functional model of the system and set of potentially impacted SIG elements. One representation of the functional model can be UML diagrams. The functional change is usually implemented at code level or design level.

The ability to understand the impact of change in UML model helps developers to evaluate the impact of change before implementing it in the code [24].

3. Goal Analysis

The goal analysis is sub-divided into two sub-phases Contribution re-analysis and Goal re-evaluation. During contribution re-analysis the impact of change is propagated throughout the SIG to evaluate its effects on system wide goals [24].

According to [24] during goal re-evaluation each operationalization is examined to determine how the proposed change influences the satisfaction of its parent goal. Any parent goal that no longer satisfied is also re-evaluated to determine how it impacts its own parent. This process is continued until all potential goals have been re-evaluated.

The output of this phase is an impact analysis report that identifies all goals that are either positively or negatively affected by the proposed change [24].

4. Decision Making

Decision making phase is sub-divided into two sub-phases decision and impact evaluation. During decision sub-phase stakeholders examine the impact report to decide whether to proceed with the proposed change or not [24]. In the impact evaluation sub-phase stakeholders evaluate the impact of the proposed change upon NFR goals and identify risk mitigation strategies [24].
Empirical Evidence

Article [24] reports an experimental evaluation of GCT in the Ice Breaker System. This system manages de-icing services to prevent ice formation on roads. This system receives information from weather stations and road servers. This system consists of 180 functional requirements and nine NFR related to accuracy, availability, safety, usability, security, extensibility, completeness and cost.

The results of the case study reveal that Goal Centric Traceability provides support to developers to manage the impact of functional change upon non-functional requirements. The use of probabilistic retrieval algorithm reduces tracing efforts in link retrieval. The probabilistic retrieval algorithm dynamically retrieves traceability links for NFR.

![Figure 13: Overview of GCT process [24]](image)

3.4.16 Analysis

On the basis of extracted information we classify requirements traceability techniques on the basis of requirements type (functional/non-functional/or both) and traceability aspects.

Value Based Requirement Traceability and Feature Oriented Requirement Traceability techniques are used to trace functional requirements. Whereas keywords & ontology, design pattern, and goal centric traceability techniques provides traceability for non-functional requirements only.

The techniques which provide traceability for both functional and non-functional requirement are Pre-RS Requirement Traceability, Event Based Traceability, Information Retrieval, Traceability Matrices, Process-Centered Environment, Aspect weaving, Scenario-Based Traceability, Hypertext-Based Approach, Rule-Based Approach.

Among all these traceability techniques only Pre-RS Requirement Traceability technique emphasis on pre-RS traceability aspects, rest of the techniques focus on post-RS traceability aspects. These results of this discussion are summarized below in Table 17.
<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Ref. No.</th>
<th>Traceability Technique</th>
<th>Aspect of Traceability</th>
<th>Type of Requirements</th>
<th>Empirical Evidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[21]</td>
<td>Value-based Requirement Traceability (VBRT)</td>
<td>Post-RS Traceability</td>
<td>Functional Requirements</td>
<td>Case study at Siemens Austria</td>
</tr>
<tr>
<td>2</td>
<td>[28]</td>
<td>Feature-Oriented Requirement Tracing (FORT)</td>
<td>Post-RS Traceability</td>
<td>Functional Requirements</td>
<td>A Case study</td>
</tr>
<tr>
<td>5</td>
<td>[8,24,25, 36,55,57,60,64]</td>
<td>Information Retrieval (IR)</td>
<td>Post-RS Traceability</td>
<td>Functional &amp; Non-Functional Requirements</td>
<td>Implemented in RETRO &amp; Case study</td>
</tr>
<tr>
<td>6</td>
<td>[8,31]</td>
<td>Rule-Based (RB) Approach</td>
<td>Post-RS Traceability</td>
<td>Functional &amp; Non-Functional Requirements</td>
<td>No Empirical Evidence Reported</td>
</tr>
<tr>
<td>13</td>
<td>[25]</td>
<td>Keywords &amp; Ontology</td>
<td>Post-RS Traceability</td>
<td>Non-Functional Requirements</td>
<td>No Empirical Evidence Reported</td>
</tr>
</tbody>
</table>
It is obvious from Table 17 that total of 15 techniques were identified in the systematic review. It is interesting to know that only four techniques have empirical evidence whereas 11 techniques have no empirical evidence.

**Figure 14: Statistical representation of the empirical evidences of requirements traceability techniques**

Similarly it is can be observed from Table 17 that some techniques are discussed in more than one papers. There are 18 papers which discuss the requirements traceability techniques whereas there are 15 traceability techniques. So there is an overlap of papers discussing the same technique as shown in Figure 15.

**Figure 15: Statistical representation of number of papers presenting the same technique**

Among the techniques shown in Figure 15 only IR and Goal centric traceability has empirical evidence. Where as the techniques like EBT, Rule based traceability and Scenario based traceability have no empirical evidence reported in literature. It is interesting to know that EBT is discussed in nine articles and it has no empirical evidence.
The empirical evidence means experiment, survey or case study conducted either in academic environment or in industry for a particular technique. In addition the results of that empirical evidence are reported in respective article.

Article [62] reports an example of M-Net, a web based conferencing system with 300 requirements and with initial set of 250 links. This example was used to illustrate the concept of EBT. This example demonstrated the feasibility of EBT to solve synchronization problems related to the updating artefacts and resolving traceability links. The author in [62] claims that the long-term empirical study of EBT is currently under way.

Article [5] describes a framework known as TraCS. It uses EBT to trace performance related requirements. There is no empirical evidence of TraCS framework. Therefore these evidences justify the fact that there is no reported empirical evidence of EBT.

The conclusions drawn from the empirical evidences of requirements traceability are reported in Table 17. This represents the results obtained by the empirical evidences of traceability techniques. These conclusions could of interest while selecting a particular technique for providing solutions in academia or industry or developing a framework for requirements traceability.

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Ref. No.</th>
<th>Traceability Technique</th>
<th>Conclusions</th>
</tr>
</thead>
</table>
| 1       | [21]     | Value-based Requirement Traceability (VBRT) | ● The traceability efforts are reduced by focusing on most important requirements as compared to full tracing.  
● It is easier to capture traceability related information in earlier phases of software development lifecycle.  
● The prioritization step of VBRT identifies important requirements to be traced in more detail than others.  
● In VBRT important requirements are identified based on parameters stakeholder value, requirements risk/volatility and tracing costs.  
● Tracing requirements into code at method level provides more useful and detailed information than tracing into class level. |
| 2       | [28]     | Feature-Oriented Requirement Tracing (FORT) | ● FORT provides variability information based of feature modelling which is useful to estimate requirements change.  
● FORT reduces efforts by feature prioritization based. |
| 3       | [8,24,25, 36,55,57,60,64] | Information Retrieval (IR) | ● It is very hard to maintain links in constantly evolving systems. IR methods facilitate dynamic link generation.  
● According to [64] the results of case study show that “IR provides a practicable solution to the problem of semi-automatically recovering traceability links between code and documentation”. |
● GCT helps to manage critical system qualities such as safety, security, reliability, usability and performance. |
Based on the systematic review results the traceability techniques can be divided into two types based on two aspects of traceability.

1. Techniques facilitating pre-RS traceability.
   This type includes those traceability techniques which help to describe the life of requirements when they are not included in the requirements specification. There is only one technique in this category which is pre-RS requirements tracing discussed in Section 3.4.3.

2. Techniques facilitating post-RS traceability.
   This type includes those techniques which help to trace the life of requirements when they are included in the requirements specification and forward. These techniques can be further divided into three types based on the systematic review results (see Table 17 Section 3.4.16). These are described below.

   1. Techniques favouring traceability of functional requirements
      These techniques support traceability of only functional requirements. There are only two techniques in this category. These are VBRT and FORT.

   2. Techniques favouring traceability of non-functional requirements
      These techniques support traceability of only non-functional requirements. The techniques included in this category are Design pattern, Keywords & ontology and Goal centric traceability.

   3. Techniques favouring traceability of both functional and non-functional requirements.
      These techniques support traceability of both functional and non-functional requirements. The techniques included in this category are EBT, IR, Hyper text based approach, Feature model based approach, Scenario based traceability, Process centric environment, Matrices and Aspect weaving.

3.5 Answering the Research Questions Stated in Review Protocol

In this section research questions addressed in review protocol are addressed.

SRQ1. What is requirements traceability based on state-of-the-art research?

The definition of requirements traceability based on state-of-the-art research is discussed in Section 3.1. It is obvious form the systematic review results that Gotel & Finkelstein, Hamilton & Beeby and ANSI/IEEE standard 830-1998 provide comprehensive and state-of-the art definitions of requirements traceability. But the definition of Gotel & Finkelstein is most widely used in literature.

Hence according to the Gotel & Finkelstein view, requirements traceability is the ability to follow and describe the life of requirements both in forward and backward direction throughout the development, deployment and refinement cycle.

SRQ2. What are the challenges/problems in implementation of requirements traceability and how did research address these problems?

The challenges / problems in implementation of requirements traceability and their solutions are discussed in Section 3.2.

The authors have classified these challenges/issues into three categories as shown in Figure 16. These are issues solved by academia, issues solved by industry and unsolved issues in
academia & industry. Among these issues the unsolved issues are of interest to the research community. The major issues in this category are for example, what is the right kind of traceability for a particular project? how much traceability is enough?

**Figure 16: Requirements Traceability Issues Based on Systematic Review**

SRQ3. What are the various requirements traceability tools?

The requirements traceability tools identified by the systematic review are discussed in Section 3.3. The authors have categorized requirements traceability tools into three categories as shown in Figure 17. These categories are requirements management tools providing traceability support, requirement traceability tools and other requirements management/traceability tools.

**Figure 17: Requirements Traceability Tools Based on Systematic Review**

SRQ4. What are the various requirements traceability techniques?

The requirements traceability techniques reported by the articles of systematic review are discussed in Section 3.4. The authors have grouped traceability according to two aspects of requirements traceability as shown in Figure 18. It is interesting to know that there is very less research work done in the area of pre-RS traceability.
Figure 18: Requirements Traceability Techniques Based on Systematic Review
4 **INDUSTRIAL INTERVIEWS**

In order to determine the requirements traceability issues interviews were conducted with two software development companies. These interviews are summarized below.

4.1 **Company A**

4.1.1 Introduction

Company A is a Sweden based world-leading telecommunication organization. It has its origin back to 1876. Company A provides telecommunications equipment and other telecom related services in 140 countries. It keeps huge share of the market. More than 1000 networks use Company A’s products and 40 percent of all mobile calls are made through its systems. Company A is one of the few telecom organizations that offer end-to-end solutions for all major mobile communication standards. Company A heavily focuses on R&D activities and promotes open standards and systems. It is depicted by the fact that over 20,000 patents are owned by Company A. Every year approximately 500 patents are filed by company A. Furthermore company A is ISO 9000 certified.

4.1.2 Interviewee

The authors contacted persons who are working specifically in the field of requirement traceability. The authors’ first interviewee was person A. She is working as discipline driver for the whole product unit. She is also responsible for the implementation of requirement traceability practices. Person A is working at company A for last 10 years and she is specifically working with requirement traceability for last 5 years. The authors’ second interviewee was person B. She is working as requirement manager and responsible for over all requirement engineering activities. Person B is working at company A for last 9 years and she is specifically working with requirement traceability for last 1.5 years.

4.1.3 Requirements Traceability

Company A believes that requirement traceability is an important and useful activity within requirement management. There are thirty people working in requirements engineering and 3-5 people are dedicated for requirements traceability practices. According to interviewees traceability is beneficial for the company due to the reasons given below:

One of the major benefits is customer satisfaction. The customer can check whether the product development undergoes according to his requirement or not. Requirements traceability provides guarantee to requirements engineers and product managers that every requirement is implemented. Requirements traceability helps to ensure impact analysis and derivation analysis. Requirements traceability facilitates impact analysis. Whenever a change request (CR) is initiated at any stage we can trace the requirements or design artefacts or test cases that can be affected. Similarly in coverage analysis requirements traceability ensures implementation of all requirements. Interviewees also mentioned that normally there are 200 change requests per project.

4.1.4 Tool Support for Traceability

Company A use an automated requirement management tool named as MARS which developed by IBM particularly for the company A. There is a specific module in the MARS which is responsible for traceability. Company A also used another tool named as ‘Focal Point’ for storing the elicited requirements. After entering the requirements into focal point ‘Main Requirement Specification (MRS)’ is generated from it. After generating MRS, it is entered into the MARS.
By using MARS, requirements are stored in a central repository. Some unique identifiers are assigned to each requirement such as ID, slogan and description. Traceability of requirements is obtained by using MARS and Focal Point as both tools are synchronized. Requirements traceability matrices (RTM) are generated to cater for requirements traceability. These matrices are used to know the status of the requirement, whether the requirement is implemented or not. RTM is extracted from MARS. MARS also facilitates change management. Change request is initiated against the requirements stored in MARS. Change control board analyzes the change request and decides to accept it or reject. In case of acceptance changes are made permanent in MARS. Versioning is also handled by the MARS.

4.1.5 Factors Influencing Requirements Traceability
According to the interviewees traceability is very well implemented using MARS. Despite this there are few issues such as the manual decomposition of master requirements specification (MRS) into detailed requirements for requirements traceability. Furthermore the maintenance cost of MARS is also very high therefore, the company A is thinking to shift on some other automated tool.

4.1.6 Future Considerations
In future company A is thinking to shift to another automated requirement management tool named Requisite Pro.

The interview questionnaire used to conduct industrial interview in company A is given in Appendix 2.

4.2 Company B
4.2.1 Introduction
The company B develops both bespoke and market-driven products for its customers. This company is involved in developing application suite and framework for mobile phones based on Symbian operating system. The company B is neither ISO9000 nor CMMI certified.

In company B the requirements engineering activities are monitored by the department named ‘software development organization’. The role of software development organization is to capture, clarify, process and trace the requirements for testing and implementation. There are thirty people work in ‘software development organization’ out of which thirteen people work in requirements capture part.

4.2.2 Interviewee
Our interviewee person C is working as a software quality engineer in company B for last two years. The person C holds MSc (Software Engineering) and is Licentiate in Software Engineering and Process Improvement from Blekinge Tekniska Högskola (BTH), Karlskrona Sweden. The current job responsibilities of person C include process development, matrix measurement and controlling the development projects.

Although person C is not directly working with requirements engineering but as a software quality engineer, he has adequate knowledge of requirements engineering activities in company B. In addition he has the academic background with some viewpoint of how it is applied in practice in industry.

4.2.3 Traceability Practices
In company B the high level requirements are known as ‘Market Requirement Specification (MRS)’. They are decomposed into ‘Product REQUIREments (PREQ)’. The Product REQUIREments (PREQ) are further decomposed into ‘Feature Specification (FS)’ which is the low level detail. Traceability is maintained by establishing connection between every MRS, PREQ, and FS and vice versa.
There is no traceability in design and source code. In testing each Feature Specification (FS) is connected to at least one test case which must have passed. The traceability is only maintained within requirements and testing phases. The requirements attributes such as feature description, dependencies with operating system, feature specification estimates are considered in traceability.

4.2.4 Tool Support for Traceability

The company B is using two different tools for managing requirements and test cases. These tools are ‘Requirements Management System (RMS)’ and ‘Test Management System (TMS)’. The RMS is used to manage requirements like MRS, PREQ and FS. It also provides traceability support but only with requirements part. TMS is used to manage and store test cases. There is no linkage between Requirements Management System (RMS) and Test Management System (TMS). The traceability between Functional Specification (FS) to test cases is maintained manually with the help of Requirements Traceability Matrix (RTM). The Requirements Traceability Matrix (RTM) is maintained in MS excel for maintaining manual traceability.

![Diagram of Requirements Management System (RMS) and Test Management System (TMS)](image)

**Figure 19: Requirements Traceability in Company B**

4.2.5 Benefits of Implementing Traceability

In company B requirements are decomposed into Market Requirements Specification (MRS) which is further decomposed into PREQ and then to FS. Traceability provides better level of details to the software engineers. Traceability helps to identify the source of feature Specification (FS). Similarly traceability from MRS to test cases ensures that the feature is complete.

Traceability helps in impact analysis. In company B impact analysis is done in several ways. If it is done early in the development life cycle during PREQ part, it is usually done by the product manager. The product manager should have sufficient competency to declare that this change will affect the product in one way or another. But during development impact analysis is done by the team responsible for managing the component affected by the change.

4.2.6 Factors Influencing Requirements Traceability

The company B has good traceability support between feature specifications, product requirement specifications and functional specification. But it is very difficult to determine requirements completion in a project. The requirement completion means that the Feature Specification (FS) have been developed and verified for specific project. There is no direct link between RMS and TMS. The FS that are verified for particular project are maintained manually using MS excel. Therefore, tool support is one factor affecting requirements traceability.

The company B is not maintaining traceability between requirements to design and code phase. The traceability is maintained only between requirements and test cases. Therefore it is very difficult to identify the components that are affected by change request in design and development phase. There are normally 5-8 change requests for every twenty features.
The interview questionnaire used to conduct industrial interview in company B is given in Appendix 3.

4.3 **Analysis of Interviews**

The results of the interviews are interesting in the sense that the views of both companies about traceability are opposite. The analysis of the interview results reveals that both companies can be placed into two different categories on the basis of traceability practices. This kind of categorization is already made by Ramesh [22] as high-end traceability users and low-end traceability users. Based on this categorization company A can be placed in the first category whereas company B can be placed in second category.

According to Ramesh [22], low-end traceability users lack a formal methodology for traceability and their goals can be accomplished by simple traceability techniques. Whereas, high-end traceability users define traceability templates and specify formal methods in CASE tools to maintain traceability.

For requirements management company A is using a formal tool MARS which also provides traceability support. MARS is specifically developed by IBM for company A. On the other hand company B lacks formal methods for traceability. The company B is using automated tools for requirements management RMS and test case management TMS. But the automated traceability is only maintained in RMS. There is no traceability in design and code. Further more to establish traceability link between RMS and TMS, traceability matrices are manually maintained using MS excel.

Company B is using static methods like traceability matrices that do not get updated as system evolve. Therefore, manual traceability matrices lose most of their usefulness after creation. The low-end traceability users exhibit this property. Whereas, high-end traceability users recognize the need for maintaining dynamic traceability that reflects the current status of the system and generate traceability documentation at any point in the development life cycle. This practice is exercised by company A.

According to Ramesh [22] low-end traceability users view traceability as a mandate from the customer while high-end traceability users consider traceability as an important component of quality system engineering process. In company A traceability is the integral part of quality assurance process. Similarly there are approximately 200 change request per project in company A. Therefore they rely on traceability to identify the affected software artefacts due to change request. Their automated requirements management tool named MARS provides richer traceability support. On the other hand company B implements traceability to ensure that two conditions are fulfilled. First is that high level requirements known as ‘Market Requirement Specification (MRS)’ are decomposed into low level requirements called ‘Feature Specifications (FS)’ and second is to guarantee that a feature is complete. In the practices followed by company B, a feature is said to be complete when there is a traceable link between MRS and test cases.

Besides the above classification we conclude that company A has well defined traceability processes. In addition to it IBM is providing them maintenance support for MARS. In future company A is planning to shift to another requirements management tool called Rational Requisite Pro. This tool is also developed by IBM and is briefly discussed in section 3.3.2.

In company B traceability practices are not fully implemented. Although they are using tool for managing requirements called RMS and tool for managing test cases known as TMS. These tools provide traceability but only for requirements and test cases. But there is no traceability support between RMS and TMS. Similarly no traceability information is maintained for traceability between design and development phase.

High-end traceability users like company A view traceability as one of the core requirements engineering practice. They believe that traceability delivers more than it costs. They can develop traceability tools in-house or can purchase automated traceability tools. On the other hand low-end traceability users like company B consider the importance of traceability but they treat traceability as an expensive overhead. Therefore, there is a need to develop a general framework which is beneficial for low-end traceability users.
The traceability related issues in Company A and Company B are listed in Table 19.

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Issues Reported in Company A</th>
<th>Issues Reported in Company B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Maintenance cost of requirements management tool MARS.</td>
<td>Manual traceability between RMS and TMS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lack of traceability between design and code.</td>
</tr>
</tbody>
</table>

In the systematic review the authors’ identified the issues related to traceability reported in Table 6 (see Section 3.2). Now the views of company A and company B are summarized in Table 20 on the issues identified by the systematic review. This provides an insight about industry view on these reported problems.

Table 20: Views of Company A & B about the issues identified in systematic review

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Ref. No</th>
<th>Issues Identified in Systematic Review</th>
<th>View of Company A &amp; Company B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[6, 62]</td>
<td>Informal development methods, Insufficient resources, Time and cost for traceability, Lack of coordination between people, Failure to follow standards.</td>
<td>Among these issues cost for traceability is a significant factor for both companies.</td>
</tr>
<tr>
<td>2</td>
<td>[4]</td>
<td>How much traceability is enough?, What is the right kind of traceability for a particular project?</td>
<td>Still unanswered.</td>
</tr>
<tr>
<td>3</td>
<td>[59]</td>
<td>Requirements change by user, Less appropriate information is available for making decision with requirements.</td>
<td>Requirements change by user is always there but both companies have maintained enough information to make decision about requirements.</td>
</tr>
<tr>
<td>4</td>
<td>[51]</td>
<td>Requirement traceability does not offer immediate benefit to development process.</td>
<td>Both companies disagree with it. Requirements traceability offers benefits for both the companies.</td>
</tr>
<tr>
<td>6</td>
<td>[5]</td>
<td>Lack of coordination between team members, Developers think that traceability costs more than it delivers, Excessive use of traceability generate more links which are not easy to manage.</td>
<td>Both companies believe that traceability delivers more than it cost. Both companies do not consider lack of coordination between team members and generation of traceability links as factors affecting traceability.</td>
</tr>
<tr>
<td>Sr. No</td>
<td>Ref. No</td>
<td>Issues Identified in Systematic Review</td>
<td>View of Company A &amp; Company B</td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td>--------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>7</td>
<td>[11]</td>
<td>Some problematic questions are identified as issues like who identify a requirement and how?, who was responsible for the requirement in start and who is responsible currently, who will take care for change(s) in requirements, what will be the effect on project in terms of knowledge loss, if some individual or the whole group leave from company.</td>
<td>These issues are not important for both companies. Company ‘A’ is using MARS and company ‘B’ is using RMS that takes care of these issues.</td>
</tr>
<tr>
<td>8</td>
<td>[58]</td>
<td>Requirements management issues in industrial projects like inadequate impact analysis, lack of information transfer.</td>
<td>Both companies have resolved these issues by requirements management tools like MARS and RMS.</td>
</tr>
<tr>
<td>9</td>
<td>[22]</td>
<td>Organizational, environmental and technical factors.</td>
<td>These factors are observed in both companies. Based on these factors Company ‘A’ is classified as “high-end traceability user” and Company ‘B’ is classified as “low-end traceability user”</td>
</tr>
<tr>
<td>10</td>
<td>[21]</td>
<td>Cost/Effort related to requirements traceability.</td>
<td>Cost related to requirements traceability is also important issue for both companies.</td>
</tr>
<tr>
<td>11</td>
<td>[25]</td>
<td>Failure to trace Non-Functional Requirements (NFR) like security, performance and usability.</td>
<td>These factors are important for both companies. The NFR like security, performance and usability are of interest for Company ‘B’ as they are developing applications for mobile phones. However Company ‘B’ lacks support for traceability of NFR.</td>
</tr>
<tr>
<td>12</td>
<td>[30]</td>
<td>Problems associated with tracing back to their sources.</td>
<td>This issue is resolved in Company ‘A’ by using MARS. MARS is an automated requirements management tool providing traceability support. This problem is important and unsolved in Company ‘B’.</td>
</tr>
</tbody>
</table>

Table 20 contains some very interesting results. For instance both companies A and B believe that traceability delivers more than it costs. But in case of company B there is no traceability in design and code phase (see Section 4.2.3). Similarly company B uses manual RTM for maintaining traceability links. In addition cost and efforts related to requirements traceability are important factors for both the companies.

Based on these results we have deduced the most important or critical issues related to requirements traceability. The source of these issues is the systematic review results and industrial interviews. These are reported in Table 21 below.
Table 21: Most Important Problems/Issues Deduced From Systematic Review and Industrial Interviews

<table>
<thead>
<tr>
<th>Sr. No</th>
<th>Problems/Issues</th>
<th>Motivation</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cost and efforts related to requirements traceability.</td>
<td>Cost and efforts are the major factors due to which companies do not use traceability.</td>
<td>Systematic review results &amp; Industrial interviews</td>
</tr>
<tr>
<td>2</td>
<td>How much traceability is enough?</td>
<td>This question is still unanswered and research community is working to solve this issue.</td>
<td>Systematic review results</td>
</tr>
<tr>
<td>3</td>
<td>Change requests or requirements change.</td>
<td>If a company is not following traceability practices then it is very difficult to manage change requests.</td>
<td>Systematic review results</td>
</tr>
<tr>
<td>4</td>
<td>Failure to trace non-functional requirements.</td>
<td>There are number of interdependencies and trade-offs between non-functional requirements. Therefore NFR are important with respect to traceability.</td>
<td>Systematic review results</td>
</tr>
<tr>
<td>5</td>
<td>Tracing requirements back to their sources.</td>
<td>Tracing requirements back to their sources helps to identify origin of requirements. This issue is one of the key factors responsible for defective requirements traceability.</td>
<td>Systematic review results</td>
</tr>
<tr>
<td>6</td>
<td>Traceability support between all phases of Software Development Life Cycle (SDLC).</td>
<td>Traceability should be maintained between all the artefacts produced during the software development and maintenance. In company ‘B’ there is a lack of traceability support design and code phase (see Section 4.2.6)</td>
<td>Industrial interviews</td>
</tr>
</tbody>
</table>

It is evident for Table 21 that some of the issues identified by the systematic review also hold in practice. In our case only two companies were interviewed and this number is very small. If the number of companies interviewed is larger than two then there is a great probability that the issues identified by systematic review also hold true in the industrial practice.

The issues given in Table 22 will be addressed by the proposed framework. These five issues have been selected based on the systematic review results, industrial interviews and after negotiation with the interviewees.

Table 22: Issues for the Framework

<table>
<thead>
<tr>
<th>ID</th>
<th>Issue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issue-1</td>
<td>Cost and efforts related to requirements traceability.</td>
</tr>
<tr>
<td>Issue-2</td>
<td>Change requests or requirements change.</td>
</tr>
<tr>
<td>Issue-3</td>
<td>Failure to trace non-functional requirements.</td>
</tr>
<tr>
<td>Issue-4</td>
<td>Tracing requirements back to their sources.</td>
</tr>
<tr>
<td>Issue-5</td>
<td>Traceability support between all phases of Software Development Life Cycle (SDLC).</td>
</tr>
</tbody>
</table>
5 A FRAMEWORK FOR REQUIREMENTS TRACEABILITY (TLFRT)

Based on the systematic review and industrial interviews we are proposing a framework for requirements traceability called ‘Three Level Framework for Requirements Traceability (TLFRT)’. The TLFRT takes into account the key factors mentioned in Table 22. TLFRT provides guidelines to a process that can be tailored to fit the needs of the high-end traceability users and low-end traceability users as well.

TLFRT is divided into three levels. These levels provide a blueprint as how to implement requirements traceability practices. The level 0 handle pre-RS traceability while level 1 and level 2 deals with the post-RS traceability. The level 1 provide traceability for the functional requirements, where as level 2 provide traceability for non-functional requirements.

Figure 20: Three Level Framework for Requirements Traceability (TLFRT)
In Figure 20 the arrows represent the types of traceability (see Section 1 for details on types of traceability).

5.1 Level 0: Pre-RS Traceability

The first step of TLFRT is based on [30], but we have added MRS to initial capabilities, PREQ to optimized capabilities and FS to finalized capabilities to tailor it to our needs.

The Pre-RS traceability is done using the approach of “Pre-RS Requirements Tracing” discussed in Section 3.4.3. The Pre-RS Requirements Tracing defined in [30] is derived from Capabilities Engineering (CE). Capabilities engineering is the process of developing change-tolerant systems based on functional abstractions called capabilities.

The approach of Pre-RS Requirements Tracing is based on three spaces namely problem space, transition space and solution space. In the problem space the user needs are decomposed into directives using Functional Decomposition graphs (FD graphs). Directives can be classified as requirements with context information or detailed characteristics of the system [30]. There are two purposes of directives. First they help to capture domain information and second, they link problem space and transition space as shown in Figure 21.

![Figure 21: Pre-RS Requirements Tracing (Level 0)](image)

In the transition space initial capabilities are identified from directives by the activity of formulation. In company ‘A’ and company ‘B’ the user needs are converted into Market or Main Requirements Specification (MRS). Basically MRS are abstract requirements. This can be seen in Figure 21 where abstract requirements like MRS are classified as Initial capabilities. Then these Initial capabilities are converted into Optimized capabilities which are subject to schedule and technology constraints [30]. These Optimized capabilities resembles with the Product REQuirements (PREQ) discussed in Section 4.2.3. Capabilities link the transition space to solution space. The directives carry domain information which remains preserved in the transition space as shown in Figure 21. Therefore directives and capabilities help to connect problem space and solution space.

In the solution space the Optimized capabilities are transformed into Finalized capabilities or requirements. These requirements are subject to quality constraints. In company ‘A’ and
company ‘B’ the Functional Specification (FS) can be treated as finalized capabilities as shown in Figure 21. This whole process explains “forward-to traceability”.

Similarly the finalized capabilities can be traced back to optimized capabilities, initial capabilities, directives and needs favouring “backward-from traceability”. This enables tracing of requirements back to their sources.

The problem space represents the view of the system from user’s perspective while transition space and solution space represent the view of the system from systems perspective.

Table 23: User and System Views in Level 0

<table>
<thead>
<tr>
<th>Name of the Space</th>
<th>View</th>
<th>Entities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem Space</td>
<td>User View</td>
<td>Needs, Directives</td>
</tr>
<tr>
<td>Transition Space</td>
<td>System View</td>
<td>Initial Capabilities /MRS, Optimized Capabilities/PREQ</td>
</tr>
<tr>
<td>Solution Space</td>
<td>System View</td>
<td>Requirements / FS</td>
</tr>
</tbody>
</table>

In order to understand traceability at Level 0 consider the example of Library Information System given below. This example will be followed throughout the TLFRT from Level 0 to Level 1 and Level 2.

5.1.1 Example

The Library Information System should be available within the library premises (Local system) and online (Web-based system). It should only provide access to the authorized members. These authorized members can use the system online and in the library. The web-based system should be efficient to process the user query within five seconds whereas as the local system should be efficient enough to process the user request in one second. The users can search, issue and reserve the books with the help of proposed system. But only Librarian should have the rights to add or update a book. The book should be issued for seven days only. If a member does not return a book after this period, the system should issue overdue notice to the member. After library hours only users with administrative rights can use the local system.

Now we apply the guidelines for Level 0 to above mentioned example and identify the corresponding entities as shown below.

<table>
<thead>
<tr>
<th>Need</th>
<th>The library information system should only allow authorized people to login.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directive</td>
<td>Provide a user login module which only allows authorized users to use the system.</td>
</tr>
<tr>
<td>Capability Initial Capability (MRS)</td>
<td>User login module.</td>
</tr>
<tr>
<td>Capability Optimized Capability (PREQ)</td>
<td>User related information should be saved in a database.</td>
</tr>
<tr>
<td>Requirement (FS)</td>
<td>User Login.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Need</th>
<th>The library information system should provide facility to search books to the authorized users.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directive</td>
<td>Provide a search module which only allows authorized users to search books using the proposed system.</td>
</tr>
<tr>
<td>Capability Initial Capability (MRS)</td>
<td>Search facility for web-based users and local users in library.</td>
</tr>
<tr>
<td>Capability Optimized Capability (PREQ)</td>
<td>The search query should return the matching records form the database.</td>
</tr>
<tr>
<td>Requirement (FS)</td>
<td>Search Book.</td>
</tr>
</tbody>
</table>

^ Example taken from the requirements elicitation assignment in the course of requirements engineering at BTH.
<table>
<thead>
<tr>
<th>Need</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The library information system should provide facility only to the</td>
<td>Librarian to add/update books.</td>
</tr>
<tr>
<td>Librarian</td>
<td></td>
</tr>
<tr>
<td>Add/update record module should be accessible only to the</td>
<td>Librarian.</td>
</tr>
<tr>
<td>Librarian</td>
<td></td>
</tr>
<tr>
<td>The books should only be issued to the registered members.</td>
<td>Directive: A book should only be issued if it is available.</td>
</tr>
<tr>
<td>Optimized Capability (PREQ): The information regarding the issued</td>
<td>books should be maintained in an issue log or database.</td>
</tr>
<tr>
<td>books should be maintained in an issue log or database.</td>
<td>Requirement (FS): Issue Book.</td>
</tr>
<tr>
<td>The members can reserve the books using Library Information System.</td>
<td>Directive: A book should be reserved only if it is issued to another member.</td>
</tr>
<tr>
<td>Optimized Capability (PREQ): A reserve queue should be maintained</td>
<td>for each reserved book. When book becomes available it should be issued to the first member in the queue.</td>
</tr>
<tr>
<td>for each reserved book. When book becomes available it should be</td>
<td>Requirement (FS): Reserve Book.</td>
</tr>
<tr>
<td>issued to the first member in the queue.</td>
<td></td>
</tr>
<tr>
<td>The system should issue overdue notice if a member does not return</td>
<td>Directive: Email overdue notice.</td>
</tr>
<tr>
<td>book within one week.</td>
<td>Capability: Initial Capability (MRS): When a book is issued to a member then day count should start that should be incremented after every 24 hours.</td>
</tr>
<tr>
<td>Optimized Capability (PREQ): When the day count exceeds seven days</td>
<td>Requirement (FS): Issue Overdue Notice.</td>
</tr>
<tr>
<td>over due notice should be emailed to the member.</td>
<td></td>
</tr>
<tr>
<td>The local system should perform a task within one second.</td>
<td>Directive: The local system should be efficient enough to perform the task in one second.</td>
</tr>
<tr>
<td>The user request should not be delayed.</td>
<td>Capability: Initial Capability (MRS): The user request should not be delayed.</td>
</tr>
<tr>
<td>Local system should offer optimized performance.</td>
<td>Requirement (FS): Performance of Local System.</td>
</tr>
<tr>
<td>The web based system should perform a task within five second.</td>
<td>Directive: The web based system should be efficient enough to perform the task in five second.</td>
</tr>
<tr>
<td>The user request should not be delayed.</td>
<td>Capability: Initial Capability (MRS): The user request should not be delayed.</td>
</tr>
</tbody>
</table>
Table 23: Requirement (FS) Performance of Web-based System

<table>
<thead>
<tr>
<th>Requirement (FS)</th>
<th>Performance of Web-based System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Need</td>
<td>The local system should be available within the library hours.</td>
</tr>
<tr>
<td>Directive</td>
<td>The local system should not facilitate users after library hours. However users with administrative rights are exempted.</td>
</tr>
<tr>
<td>Capability</td>
<td>Initial Capability (MRS) Time dependent availability of local system.</td>
</tr>
<tr>
<td></td>
<td>Optimized Capability (PREQ) After library users with the administrative rights can use the local system.</td>
</tr>
<tr>
<td>Requirement (FS)</td>
<td>Availability of Local System.</td>
</tr>
</tbody>
</table>

We can map this process to Table 23 as shown below. This provides better understanding of the system by clearly differentiating between the user and system views.

Table 24: User and System Views at Level 0 of Library Information System

<table>
<thead>
<tr>
<th>Need</th>
<th>User View</th>
<th>The web Based system should be available 24/7.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directive</td>
<td>User View</td>
<td>The web based system should be efficient enough to perform the task in five second.</td>
</tr>
<tr>
<td>Capability</td>
<td>System View</td>
<td>The user request should not be delayed.</td>
</tr>
<tr>
<td></td>
<td>System View</td>
<td>Web based system should offer optimized performance.</td>
</tr>
<tr>
<td>Requirement (FS)</td>
<td>System View</td>
<td>Availability of Web Based System.</td>
</tr>
</tbody>
</table>

The output of Level 0 is software requirements specification as shown in Figure 18. The software requirements specification should preferably be documented using “IEEE std 830-1998” template. In TLFRT the functional requirements in SRS will be traced at Level 1 while NFR will be traced at Level 2.

We identify the functional and NFR in the example mentioned in Section 5.1.1.

Table 25: Functional Requirements in Library Information System

<table>
<thead>
<tr>
<th>Requirement Identifier</th>
<th>Functional Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReLib_userLogin</td>
<td>User Login</td>
</tr>
<tr>
<td>ReLib_searchBook</td>
<td>Search Book</td>
</tr>
<tr>
<td>ReLib_addupdateBook</td>
<td>Add/Update Book</td>
</tr>
<tr>
<td>ReLib_issueBook</td>
<td>Issue Book</td>
</tr>
<tr>
<td>ReLib_reserveBook</td>
<td>Reserve Book</td>
</tr>
<tr>
<td>ReLib_issueNotice</td>
<td>Issue Overdue Notice</td>
</tr>
</tbody>
</table>
Table 26: NFR in Library Information System

<table>
<thead>
<tr>
<th>Requirement Identifier</th>
<th>Non-Functional Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReLib_perfLocalSys</td>
<td>Performance of Local System</td>
</tr>
<tr>
<td>ReLib_perfWebbasedSys</td>
<td>Performance of Web-based System</td>
</tr>
<tr>
<td>ReLib_availLocalSys</td>
<td>Availability of Local System</td>
</tr>
<tr>
<td>ReLib_availWebbasedSys</td>
<td>Availability of Web-based System</td>
</tr>
</tbody>
</table>

5.1.2 Issues solved by Level 0

In this section two of the issues (see Section 4.3 Table 22) are presented with a description of how TLFRT contribute in resolving them.

**Issue - 2: Change request or Requirements change.** Level 0 of TLFRT helps to manage requirements change. In case of requirements we can trace the requirements back to the problem space. This helps to identify the issuer of requirements and the user needs that a particular requirement satisfies.

**Issue - 4: Tracing requirements back to their sources.** Level 0 of TLFRT provides facility to trace requirements back to their sources. In this case we move from solution space to problem space via transition space. Similarly it also provides support to trace user needs to requirements as shown in Figure 18.

5.1.3 Limitations of Level 0

This section describes the limitation of Level 0 of TLFRT.

**Limitation – 1: Abstract Requirements.** The Level 0 converts user needs into abstract requirements known as initial capabilities or MRS, which are then converted into optimized capabilities (PREQ) and finalized capabilities (FS). But in practice requirements can come in all shapes and sizes in companies. This problem may become serious if the requirements come as detailed requirements. In order to solve this issue and to keep requirements appropriately abstract each time Requirements Abstraction Model (RAM) proposed by Dr. Tony Gorschek can be used. The RAM consists of three steps as shown in Figure 22.

Figure 22: Requirements Abstraction Model (RAM) [13]

In RAM first step (Specify) aims at providing an overview of the raw requirements [13]. In this phase four attributes Description, Reason/Benefit/Rationale, Restrictions and Title are manually inserted to the raw requirements.

During the second step (Place) level of the requirement is analyzed. This step has four levels of requirements *Product level, Feature level, Function level* and *Component Level*.

The last step of RAM (Abstraction) involves breakdown or abstraction of requirements based on original requirements. In abstraction level requirements are compared with the product strategies and therefore abstraction of requirements is done to Product level.
Limitation – 2: Lower Level Requirements in the Beginning: The Level 0 of TLFRT may not work properly if requirements at the lower come in the beginning.

Limitation – 3: Scalability: The higher number of requirements or user needs (e.g. 10000 requirements or use needs per year) may affect proper working at Level 0.

5.1.4 Rationale for using Pre-RS Requirements Tracing

The systematic review results reveal that one of the major issues related to requirements traceability is tracing requirements back to their sources as mentioned in Table 4. In addition it is empirically established that defective requirements traceability results due to inadequate pre-RS tracing as compared to post RS-tracing [30]. The technique of Pre-RS Tracing facilitates pre-RS traceability. According to the systematic review results reported in Table 17 (Section 3.4.16), pre-RS Tracing is the only technique dealing with pre-RS traceability.

Normally in pre-RS tracing problem space and solution is present. Now much of the information is lost when we move from problem space to solution space resulting in misunderstood and inconsistent requirements. This loss of information is due to the complexity gap between these two spaces. The technique of pre-RS requirement tracing helps to overcome this complexity gap by introducing transition space between problem space and solution space.

5.2 Level 1: Post-RS Traceability with Functional Requirements

In order to trace functional requirements in our framework we use the approach of Value Based Requirements Traceability (VBRT) discussed in Section 3.4.1. The VBRT is based on five phases [21] which are briefly explained below in relation with TLFRT.

5.2.1 Requirements definition

In the requirements definition phase the functional requirements are identified from the Software Requirements Specification (SRS). The reason for this is that VBRT provides traceability with the functional requirements.

We will use the functional requirements of the Library Information System given in Table 25.

5.2.2 Requirements prioritization

In this phase normally all stake holders evaluate the requirements based on these three criteria. During requirements prioritization all the functional requirements identified in requirements definition phase are categorized based on three parameters. These parameters are value, risk and effort for each requirement.

A question may arise that why we should only consider three prospective value, risk and effort for prioritizing requirements and what are the benefits related to traceability associated with it. The motivations for prioritizing requirements based on these three prospective are given below.

The value prospective of requirements represents the importance of requirements to stakeholders. This value is measured on three-point scale (high ‘+’, medium ‘0’ and low ‘-’). The reason for this categorization is that the high value requirements should be traced in more detail as compared to medium or low value requirements as they represent core functionality of the system [21].

The risk prospective represents the volatility of requirements measured on three-point scale (high ‘+’, medium ‘0’ and low ‘-’). More risky requirements are subject to change and during this process they may need many cycles of adjustments. It is important to recognize the impact of change on system design and other software artefacts. It is worth while to trace high risk requirements because in change impact analysis these traces between high value requirements are needed frequently than traces to stable requirements [21].
The effort prospective represents the estimated amount of time and human resource required to fulfill a requirement. For instance, there is ‘n’ number of artefacts. Then in order to maintain traceability between all ‘n’ artefacts, the complexity of requirements tracing is ‘n^2’. Now efforts explode when the number of requirements increases in this scenario. This is one of the major issues or problems related to requirements traceability based on the systematic review results and industrial interviews reported in Table 19 (see Section 4.3). But VBRT takes care of this problem by prioritizing requirements based on effort using three-point scale (high ‘+’, medium ‘0’ and low ‘-’).

This prioritization step is a very crucial step because it helps to determine the priority level of requirements. The output of this step is three priority levels as shown in Table 29. The requirements at priority level 1 can be traced in more detail than the requirements at priority level 2 and priority level 3. This step is a suitable mean to identify which requirements are more valuable than others. Similarly due to this prioritization step the effort associated with requirements tracing are reduced. These efforts are reduced due to the fact that we can trace requirements at level 1 in more detail than others. A comparison of efforts reduced by VBRT and other traceability techniques is given in Table 30 (Section 5.2.8).

But we customize this phase for our framework based on the industrial case study on VBRT reported in [21]. In our case only project manager or product manager can measure value, risk and effort of each requirement where as other stakeholders can only assess the value of requirements. Now consider the example of Library Information System given in Section 5.1.1. The requirements prioritized by the project manager or product manager are shown in Table 27 and requirements prioritized by other stakeholders are shown in Table 28.

**Table 27: Requirements Prioritized By the Project Manager/Product Manager (PM)**

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Functional Requirement</th>
<th>Value (V)</th>
<th>Risk (R)</th>
<th>Effort (E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReLib_userLogin</td>
<td>User login</td>
<td>+</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>ReLib_searchBook</td>
<td>Search book</td>
<td>+</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>ReLib_addupdateBook</td>
<td>Add / Update book</td>
<td>+</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>ReLib_issueBook</td>
<td>Issue book</td>
<td>+</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>ReLib_reserveBook</td>
<td>Reserve book</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>ReLib_issueNotice</td>
<td>Issue overdue notice</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

**Table 28: Requirements Prioritized By the Stakeholders (S)**

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Functional Requirement</th>
<th>Value (V)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReLib_userLogin</td>
<td>User login</td>
<td>+</td>
</tr>
<tr>
<td>ReLib_searchBook</td>
<td>Search book</td>
<td>+</td>
</tr>
<tr>
<td>ReLib_addupdateBook</td>
<td>Add / Update book</td>
<td>0</td>
</tr>
<tr>
<td>ReLib_issueBook</td>
<td>Issue book</td>
<td>+</td>
</tr>
<tr>
<td>ReLib_reserveBook</td>
<td>Reserve book</td>
<td>0</td>
</tr>
<tr>
<td>ReLib_issueNotice</td>
<td>Issue overdue notice</td>
<td>+</td>
</tr>
</tbody>
</table>
The Table 27 and 28 are merged to get Table 29 shown below containing the prioritized requirements. In Table 29 ‘PM’ represents project manager/product manager and ‘S’ stands for stakeholders.

<table>
<thead>
<tr>
<th>Req. ID</th>
<th>Functional Requirement</th>
<th>Value (V)</th>
<th>Risk (R)</th>
<th>Effort (E)</th>
<th>Risk/Effort (RE)</th>
<th>Value (V)</th>
<th>Priority Level (L)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PM</td>
<td>PM</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>+</td>
</tr>
<tr>
<td>ReLib_userLogin</td>
<td>User login</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>ReLib_searchBook</td>
<td>Search book</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>ReLib_addupdateBook</td>
<td>Add / Update book</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>ReLib_issueBook</td>
<td>Issue book</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>+</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>ReLib_reserveBook</td>
<td>Reserve book</td>
<td>2</td>
<td>1</td>
<td>+</td>
<td>++</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>ReLib_issueNotice</td>
<td>Issue overdue notice</td>
<td>3</td>
<td>0</td>
<td>+</td>
<td>++</td>
<td>+</td>
<td>1</td>
</tr>
</tbody>
</table>

In Table 29 the third column “Value” represents votes by one or more project managers/product managers, in the example in Table 29 there are three project managers. Now we assume that 2 project managers voted for requirement ‘ReLib_userLogin’ to be important, i.e. 2 x ‘+’, one project manager voted for requirement ‘ReLib_userLogin’ to be medium ‘0’ and none of the project managers voted for requirement ‘ReLib_userLogin’ to be less important ‘-’.

The columns “R” “E” contains the value for risk and effort given by the project manager/product manager and seventh column “V” contains the value given by the stakeholders as mentioned in the Tables 26 and 27 respectively. The column “RE” contains the risk per effort for every requirement measured on the scale ‘++ (very important)’, ‘0 (medium)’ and ‘-- (less important)’.

The risk (R) for requirement ‘ReLib_userLogin’ is ‘+’ and effort (E) is ‘0’, therefore its risk per effort (RE) is ‘0’ which is medium. The column “L” contains the priority levels measured on the scale ‘1 (high priority level)’, ‘2 (medium priority level)’ and ‘3 (low priority level)’.

The RE value for requirements ‘ReLib_reserveBook’ and ‘ReLib_issueNotice’ is ‘++ (very important)’and ‘ReLib_issueNotice’ is voted as ‘+ (important)’ by two project managers where as ‘ReLib_issueNotice’ is voted as ‘+ (important)’ by three project managers. Therefore requirements ‘ReLib_reserveBook’ and ‘ReLib_issueNotice’ are at high priority level or level 1.

The VBRT case study reported in [21] is based on a system containing 46 requirements. It was estimated that average duration of requirements prioritization was 50 to 60 minutes per person.

5.2.3 Requirements packaging

This step is optional in the process of VBRT [21]. During requirements packaging the software architects can identify the clusters of requirements. These clusters of requirements can be based on the priority levels (high/medium/low) identified in the requirements prioritization step. These clusters facilitate development of architecture based on the prioritized requirements.

Although this step is important but if this step is followed it can lead to developing better software architecture based on prioritized requirements. For example consider the requirements
‘Reserve book’ and ‘Issue over due notice’ as mentioned in Table 29. These requirements are at high priority level. During requirements packaging these requirements can be clustered together. Then this cluster of requirements can be used for developing software architecture or these requirements can also be regarded as important in coding phase.

A question may arise that how requirements packaging can be relevant for TLFRT. TLFRT uses the approach of VBRT for post-RS traceability. The requirements packaging step is a part of VBRT. If this step is followed it can lead to better results like identifying critical requirements during design phase. Otherwise this step can be skipped as it is optional.

5.2.4 Requirements linking

In requirements linking the traceability links are established between requirements and other software artefacts. These software artefacts can be design documents, source code or test cases.

The high priority level requirements can be traced in more detail as compared to medium priority level or low level priority level requirements. A better approach is to use a ‘traceability plan’ specifying requirements at a particular level to be traced completely.

For example if requirements with high priority level are to be traced to design and code. Then the Requirements Traceability Matrix (RTM) should be used for keeping the traceability links between various software artefacts like requirements documents, design documents and code.

A question may arise that RTM are subject to scalability constraints and TLFRT uses RTM for traceability between requirements to design and design to code phase. There are three reasons for using RTM in TLFRT.

First is that use of RTM is also evident from systematic review. In the case study of VBRT reported in [21], RTM is used to maintain traceability links between various software artefacts. Secondly during the industrial interview in Company ‘B’, it was observed that RTM were used (see Section 4.2.4). In company ‘B’ the RTM is maintained with the help of MS Excel. Third reason for using RTM is to keep framework as simple as possible.

![Figure 23: Traceability between Various Software Artefacts Using RTM](image)

The traceability links in the RTM can be used for both forward traceability and backward traceability.

5.2.5 Evaluation

In the evaluation phase the overall process of functional requirements traceability can be evaluated. A better approach is to make a “traceability plan”. This traceability plan can be used for making various decisions. For example with the help of traceability plan we can see the impact of change for a certain requirement.
Consider ‘Issue Book’ requirement mentioned in Table 29 which is at priority level 1. If due to any reason this requirement is changed, then with the help of traceability plan we can identify the other components like design components affected due to this change.

5.2.6 Issues Solved by Level 1

This section describes how Level 1 of TLFRT helps to solve three of the issues mentioned in Table 22 (Section 4.3).

Issue - 1: Cost and efforts related to requirements traceability. The cost and effort related to requirements traceability is reduced by the prioritizing requirements based on three factors value, risk and effort (see Section 5.2.2). Then we assign three priority levels to these prioritized requirements (see Table 29 Section 5.2.2). The requirements at priority level 1 can be traced in more detail than the requirements at priority level 2 and priority level 3.

Similarly due to this prioritization step cost related to requirements traceability is also reduced. The reason for this is that we do not need to trace all the requirements. Table 30 (Section 5.2.8) provides description how efforts and cost is reduced with the help of VBRT.

The systematic review results reveal that with the help of VBRT the tracing efforts are reduced to 35% in comparison to full tracing (see Section 3.4.1).

Issue - 2: Change requests or requirements change. The Level 1 of TLFRT helps to manage requirements change. The example of ‘Issue Book’ requirement given in Section 5.2.5 helps to understand how Level 1 solves this issue.

Issue - 5: Traceability support between all phases of Software Development Life Cycle (SDLC). The Level 1 facilitates traceability between all phases of software development life cycle. But this traceability is only with the prioritized requirements. For instance the requirements at priority level 1 can be traced to design and code phase. This facilitates traceability between various software artefacts like SRS, design and code.

5.2.7 Limitations of Level 1

Limitation – 1: Scalability of RTM. One of the major weaknesses of RTM is the difficulty to maintain large number of links. The traceability between requirements to design and then design to code using RTM (see Figure 21) may not be scalable. In order to overcome this issue IR methods (see Section 3.4.5) can be used instead to RTM.

According to the systematic review results (see Table 17 Section 3.4.16) IR methods helps to generate dynamic links instead of user-defined explicit links as in the case of RTM. In IR methods a query is constructed from the keyword of the requirement to be traced. Then based on the similarity of the query with the other software artefacts, the retrieval algorithm returns set of likely links to the user [5].

5.2.8 Rationale for using VBRT

The high efforts associated with requirements tracing is one of the main reasons due to which companies do not use requirements traceability in practice. But VBRT reduces the tracing efforts without scarifying traceability benefits [21]. The traceability efforts are reduced by the prioritization step of VBRT which identifies more valuable requirements to be traced than the others.

There is another approach of Feature Oriented Requirements Traceability (FORT) discussed in Section 3.4.2 which reduces the traceability efforts by prioritization. But based on the systematic review results it is evident that VBRT is successfully implemented in an industrial case study. On the other hand FORT is used in a case study with a small car rental system. Based on the industrial empirical evidence of VBRT and its conclusions as reported in Table 18, we have chosen VBRT for tracing functional requirement in this framework.
Table 30: Comparison of Tracing Techniques [21]

<table>
<thead>
<tr>
<th></th>
<th>Proactive effort to identify and maintain traces</th>
<th>Additional effort and delay in case of change request</th>
<th>Overall cost in % to total project cost (optimistic to worst case)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ad-hoc Tracing</td>
<td>Low</td>
<td>Extremely high</td>
<td>0 % to 20 %</td>
</tr>
<tr>
<td>Full tracing</td>
<td>High</td>
<td>Low</td>
<td>5% to 15%</td>
</tr>
<tr>
<td>VBRT</td>
<td>Medium</td>
<td>Low</td>
<td>2% to 7%</td>
</tr>
</tbody>
</table>

According to the comparison reported in Table 30, it is evident that VBRT is an appropriate solution to trace requirements by reducing cost and effort.

5.3 Level 2: Post-RS Traceability with Non-Functional Requirements

The systematic review results report that one of the major issues related to requirements traceability is failure to trace Non-Functional Requirements (NFR) like performance security and usability as mentioned in Table 4. This is due to the interdependencies and trade-offs that exist between NFR [24].

In order to trace NFR, TLFRT uses the approach of Goal Centric Traceability (GCT) discussed in Section 3.4.15. The NFR mentioned in Table 26 will be traced at Level 2. The GCT is based on four phases [24] and these phases are discussed below in context with TLFRT.

5.3.1 Goal Modelling

The NFR identified in the Software Requirements Specification (SRS) and design documents are modelled as goals and operationalizations in the Softgoal Interdependency Graph (SIG). This SIG should be maintained throughout the software lifecycle.

In SIG NFR are represented as ‘goals’. These goals can be divided into subgoals. These subgoals define the qualities of the system as desired by its stakeholders. The design solution which helps to achieve a goal is known as ‘operationalization’. An operationalization can contribute positively or negatively to its parent goal or it may not have any impact on its parent goal. These contribution relationships are represented as ‘+ (helps)’, ‘++ (makes)’, ‘– (hurts)’, ‘-- (breaks)’ and ‘? (unknown)’ [24].

In order to develop SIG we do goal modelling of ‘Performance of local system’ requirement given in Table 26.
Figure 24: Goal Modelling of ‘Performance of local system’ NFR using SIG

The notations used in SIG are described below.

<table>
<thead>
<tr>
<th>Table 31: Notations used for SIG</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Cloud]</td>
</tr>
<tr>
<td>![Cloud]</td>
</tr>
<tr>
<td>![Smiley face]</td>
</tr>
<tr>
<td>![Smiley face]</td>
</tr>
<tr>
<td>![Checkmark]</td>
</tr>
</tbody>
</table>
5.3.2 Impact Detection

The impact detection aims to identify the components in design level or code level that can be affected due to the change in any NFR. In the impact detection phase traceability links are established between the functional model and elements in the SIG that can be affected by the functional change. The functional model can be UML diagrams or code [24].

The traceability links are evaluated by the stakeholders and the incorrect links are discarded.

**Figure 25: Establishing Traceability Links Between Functional Model and SIG Elements**

In order to understand these steps consider Figure 24. Suppose due to a functional change the operationalization ‘Less disk accesses’ is affected. The potentially impacted SIG elements due to this functional change are shown below.

Then traceability links should be established between the impacted SIG elements and the functional model (UML diagram or code). This helps to identify design components or code that need to be modified due to this functional change. These traceability links can be maintained by using traceability matrix.

The output of this phase is a set of directly impacted SIG elements as shown in Figure 26.
5.3.3 Goal Analysis

In the Goal analysis phase the impact of change is propagated to all the regions in the SIG that are likely to be affected by the functional change [24]. This helps to evaluate the impact of functional change to all the system goals.

Then each operationalization is re-evaluated to determine the impact of change on the satisfaction of its parent’s goals. This process of ‘goal re-evaluation’ continues until all the potentially elements in SIG are re-evaluated.

The end product of this phase is an impact analysis report that highlights the positive or negative effect of change on the NFR modelled as goals in SIG.

5.3.4 Decision Making

In the decision making phase the stakeholders can examine the “impact analysis report” for decision making. For instance the stakeholders can decide whether to continue with the
proposed change or to discard it. Software architects can evaluate the impact of change on the software architecture and developers can identify methods or classes affected due the impact of change in the code.

5.3.5  Issues Solved by Level 2

This section highlights how Level 2 to TLFRT provides solution to some of the issues identified in Table 22 (Section 4.3).

**Issue – 3: Failure to trace non-functional requirements.** The Level 2 of TLFRT facilitates the traceability of NFR. The NFR are first decomposed into elements like goals, subgoals and operationalizations. These elements are drawn in SIG as shown in Figure 22. Then the impact of change is detected as explained in Section 5.3.2, 5.3.3 and 5.3.4.

5.3.6  Limitations of Level 2

**Limitation – 1: Scalability of RTM.** The Level 2 uses RTM for maintaining traceability links between the elements affected by the functional change in SIG and functional model as shown in Figure 23. But it can be very difficult to use RTM to maintain links in continuously evolving systems.

There are two solutions to overcome this limitation based on the systematic review results. First solution is to use IR methods (see Section 3.4.5). IR methods use a dynamic retrieval algorithm that takes cares of continuously changing links. Second solution is to use the techniques of Keywords & Ontology (see Section 3.4.13). In this approach the keywords can be embedded in the operationalizations of SIG and with the UML diagrams or code. Therefore there is no need to maintain RTM.

5.3.7  Rationale for Using GCT

There are number of other techniques for managing traceability with NFR as mentioned in Table 17 (Section 3.4.16). But according to the systematic review results GCT has been successfully used in an experiment on a system comprising 180 functional requirements, nine distinct SIGs representing NFR of availability, accuracy, completeness, extensibility, cost, security, performance, safety and usability [24].

The results of this experiment show that GCT facilitates to manage the impact of functional change on NFR. Secondly GCT helps to deal with critical system qualities such as safety, security, reliability, usability and performance.
6 **STATIC VALIDATION OF THE FRAMEWORK**

This section presents a discussion about validation of the proposed framework for requirements traceability. The proposed framework was statically validated through interviews and reviews of the people, interviewed earlier for the development of the framework (Section 4).

6.1.1 Validation Design

In order to validate the framework a detailed presentation was given to the practitioners. During the presentation results of the systematic review, industrial input and framework were focused. In addition while presenting framework an example (with 10 requirements) was used to provide the clear idea of the use of framework. After presentation a discussion was made using a set of qualitative and quantitative questions regarding framework. The questions designed for discussion are given in Table 32.

An effort was made to explicitly map the questions to framework parts. Therefore questions on each part of the framework were formulated as shown below.

<table>
<thead>
<tr>
<th>Table 32: Questionnaire for Framework Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Level 0</strong></td>
</tr>
<tr>
<td>1. Do you think that TLFRT facilitates in tracing requirements back to their sources?</td>
</tr>
<tr>
<td>2. Do you think that TLFRT helps to manage change request or requirements change?</td>
</tr>
<tr>
<td><strong>Level 1</strong></td>
</tr>
<tr>
<td>3. Do you think that tracing prioritized requirements using TLFRT helps to reduce cost and effort?</td>
</tr>
<tr>
<td>4. TLFRT uses three prospective to prioritize requirements. Do you think that these prospective are enough for prioritizing requirements?</td>
</tr>
<tr>
<td>5. In TLFRT only product manager or project manager can prioritize requirements based on three prospective cost, value and effort. Where stakeholders can only prioritize requirements based on one prospective which is value. Do you think that this kind of division is appropriate?</td>
</tr>
<tr>
<td>6. In the Level 1 of TLFRT the ‘requirements packaging’ step is optional. What is your opinion that this step should be optional or mandatory?</td>
</tr>
<tr>
<td>7. TLFRT uses RTM for maintaining traceability links. Do you think that it is scalable to use RTM in projects of your company?</td>
</tr>
<tr>
<td>8. Do you think that Level 1 of TLFRT helps to trace functional requirements?</td>
</tr>
<tr>
<td><strong>Level 2</strong></td>
</tr>
<tr>
<td>9. In Level 2 of TLFRT Softgoal Interdependency Graph (SIG) is used to model NFR. Do you think that SIG is helpful to model NFR?</td>
</tr>
<tr>
<td>10. In Level 2 of TLFRT traceability links are established between set of elements in SIG affected by the functional change and functional model. This functional model can be design documents or code. In your opinion due to functional change in any NFR the</td>
</tr>
</tbody>
</table>
6.1.2 Validation Feedback from Company ‘A’

In company ‘A’ framework was presented to Person ‘A’ and Person ‘B’ whose brief introduction is given in Section 4.1.2.

The interviewees agreed that TLFRT is customizable framework. They were of the opinion that TLFRT is useful for both high-end traceability users and low-end traceability users. Initially it may take more cost but once it is developed and implemented it is quite useful.

The interviewees agreed that Level 0 of TLFRT facilitates tracing requirements back to their sources. Similarly Level 0 helps to manage change requests but still it is subject to scalability constraints.

They appreciated the use of three prospective for requirements prioritization. These three prospective are risk, value and effort. They pointed out that these parameters should not be changed during the project. Company ‘A’ is using requirements management tool for traceability so they do not construct manual RTM. Therefore the question of scalability of RTM in projects does not hold in this case. In their point of view Level 1 of TLFRT helps to trace functional requirements.

In company ‘A’ functional and non-functional requirements are traced by the requirements management tool. It was difficult for them to trace NFR by modelling them in SIG. The interviewees were not familiar with the concept of SIG. But they agreed that once you have knowledge of SIG, it is very easy to model NFR with it. In their view the impact analysis report produced during Level 2 of TLFRT is very useful for decision making. They consent that Level 2 of TLFRT helps to trace NFR.

The validation feedback from company A is given in Appendix 4.

6.1.3 Validation Feedback from Company ‘B’

In company ‘B’ framework was presented to Person ‘C’ whose brief introduction is given in Section 4.2.2.
The interviewee showed interest to use this framework when it is developed in the form of an automated tool. But as this framework is based on three techniques, so proper training is required to use it. Moreover this framework is customizable and can be used in company B.

The interviewee agreed that TLFRT is affordable for both high-level traceability users and low-level traceability users. However its initial cost may be higher. The interviewee appreciated the idea of explicitly mentioning the limitations of each level of the framework.

According to the interviewee level 0 of TLFRT facilitates tracing requirements back to their origin and forward to the requirement specification. The interviewee agreed to the fact that requirements may not necessarily come for users, they may also come for other requirements. So he appreciated the use of RAM model to handle high level and low level requirements at level 0.

The interviewee complemented the idea to prioritize the requirements and then tracing these prioritized requirements in order to reduce cost and efforts. Furthermore the three parameters value, risk and effort are enough for prioritizing the requirements for company B. In TLFRT only product manager or project manager can prioritize requirements based on three prospective cost, value and effort. Where stakeholders can only prioritize requirements based on one prospective which is value. However the interviewee pointed out that the person or department in a company who owns the budget should prioritize the requirements based on these three parameters.

The interviewee shared his personal experience of comfortably using RTM in actual practice. He appreciated the use of RTM for maintaining traceability links in TLFRT which the interviewee thinks is the cheapest solution. In order to overcome the scalability constraints associated with RTM, the interviewee appreciated the alternative of using IR methods for dynamic link generation.

The interviewee was of the opinion that if we go for a cheaper solution and low investment in beginning, then people will start using this framework. Later on in advance version of TLFRT IR methods can be used.

The interviewee agreed that level 2 of TLFRT takes care of NFR. But in order to use this framework one should have knowledge of SIG so as to model NFR. The interviewee agreed that the ‘impact analysis report’ produced during level 2 of TLFRT is useful for project managers and stakeholders.

The validation feedback from company B is given in Appendix 5.

6.1.4 Lesson Learned

Following are some of the important points about TLFRT that are deduced based on the framework validation interviews.

- The TLFRT is customizable framework and it can be easy to use when it is implemented in the form of an automated tool.
- When TLFRT will be implemented in the form of a tool then proper training is needed to use it.
- The three levels of TLFRT should not be compulsory and may not be followed in a strict sequence. A company may use level 1 to trace requirements and later on they may use level 0.
- The cost and efforts associated with requirements traceability can be reduced by tracing the prioritized requirements.
- The three parameters value, risk and effort are enough to prioritize requirements in level 1 of TLFRT.
- The person who owns the budget should prioritize the requirements based on three parameters in level 1.
- In order to keep this framework simple it is better to use RTM for maintaining traceability links. However in later version of TLFRT, IR methods can be used for automatic link generation.
The level 2 of TLFRT can take care of NFR but one should have knowledge of SIG to understand the working at this level.

The impact analysis report produced during level 2 could be of interest to project managers and other stakeholders.

6.2 Answering the Research Questions

In this section, the research questions stated in chapter 1 (Section 1.2) are addressed. In order to answer these research questions, a mixed approach using quantitative & qualitative research methodologies is used, which is illustrated in Figure 27.
Figure 27: Overview of the Methodology for Answering Research Questions
7 VALIDITY THREATS

The validity threats enhance the accuracy of research design by identifying those potential factors which can affect the results. In this section four different kinds of validity threats and their implications are discussed. These four types of threats as outlined by [9] are conclusion validity, construct validity, internal validity and external validity.

7.1 Conclusion Validity (Reliability)

Conclusion validity ensures that results of a particular study are reliable and they can lead to correct conclusion. Threats to conclusion validity are concerned with those issues that affect the reliability of results [9, 68, 69]. Conclusion validity is also known as reliability [68].

One of the main purposes of review protocol in systematic review is to eliminate researcher bias [16]. The review protocol (see section 2.1) was reviewed by two independent researchers from Blekinge Institute of Technology, Sweden having experience in conducting systematic reviews. In addition the relevance of search terms and search resources defined in review protocol was ensured by conducting pilot searches with the help of librarian at Blekinge Institute of Technology, Sweden.

The questionnaires used for the industrial interviews and framework validation were validated to rectify poor questions and flow in the layout of questionnaire.

The heterogeneity of subjects is another threat to conclusion validity [9]. Heterogeneity of subjects means that the subjects belong to varied group with respect to background, education and experience. While homogeneity of subjects means that the subjects belong to same group based on education, background and experience. The subjects of our industrial interviews are neither very heterogeneous nor very homogeneous.

The framework for requirements traceability is based on those techniques which have proved to be useful based on the systematic review results. These techniques have been used to solve problems related to pre-RS traceability and post-RS traceability.

7.2 Construct Validity

Construct validity identifies the relationship between theory and observation [69]. For instance if we are using experiment as a research methodology for a research study, then according to construct validity the results of the experiment should be generalized to the main idea behind the experiment [9].

Evaluation apprehension is the main threat to construct validity. According to [9], evaluation apprehension means that humans have the tendency to look better when they are evaluated. On the other hand some people are afraid of being evaluated. In order to eliminate this threat the interviewees in both the companies were ensured that their names and their company names will not be specifically mentioned in the thesis work.

Mono-operation bias is another threat to construct validity. Mono-operation bias means that there is a single independent variable, case, subject or treatment in a research study [9]. We have eliminated this threat by conducting interviews at two software companies. In first company we interviewed two requirement engineers and in second we interviewed one quality engineer. The proposed framework is statically validated through interviews and reviews of the people that were interviewed earlier for the development of framework.
7.3 Internal Validity (Causality)

Internal validity addresses the fact that research design should allow researcher to draw conclusions from the causes and effects [9, 68, 69]. Internal validity identifies those factors which affect independent variables without researcher’s knowledge [9, 68]. Internal validity is also known as causality [68].

The interviews were recorded. It is a common observation that people may not feel comfortable if the interview is audio taped. In order to eliminate this potential threat to conclusion validity the interviewees were guaranteed that the recordings will only be used by the thesis authors.

The internal validity threat related to systematic review is the publication bias. Publication bias refers to the fact that positive results are more likely to be published than the negative facts [16]. In order to eliminate this threat the selected conferences and proceedings mentioned in Table 1 were manually searched. Similarly the “study quality assessment and procedures” mentioned in the review protocol eliminated this factor of publication bias.

Selection of subjects from population also affects internal validity [9]. In our case we interviewed people working in the process area of requirements engineering. However one of our subjects in company ‘B’ does not work directly with requirements traceability. But he has adequate educational background, experience and knowledge about the practices related to requirements traceability and requirements engineering carried out in company ‘B’.

7.4 External Validity (Generalizability)

External validity ensures that the results of a particular research study can be generalized or not [9, 68, 69]. External validity is also known as generalizability [68].

One of the threats to external validity is the interaction of selection and treatment [9]. This occurs when wrong subjects are taken from the population, then the results of such a research work cannot be generalized to the whole population. But in our case the subjects of our interviews are people working in the process area of requirements traceability.

The proposed framework is based on the systematic review and interviews with the industry practitioners. This framework is validated in two companies that were interviewed. Both of these companies are working in the same domain i.e. telecommunications. There is a threat that validation results may not be generalized to the companies related to other software domains. However this framework provides only guidelines to implement the practices of requirements traceability. The proposed framework is customizable to the organization needs.
8 **EPILOGUE**

In this chapter the conclusions of the master’s thesis are presented. It also presents in which areas further work might be needed.

8.1 **Conclusions**

This thesis presents a systematic review of requirements traceability and a framework for requirements traceability called TLFRT. In fact these two are the major contributions of this thesis. Apart from the actual framework, the study results does not provide any evidence of the attempts to perform systematic review within requirements traceability.

The thesis has fulfilled all the requirements of a standard systematic review as proposed by the Kitchenham. The thesis presents the current state-of-the-art research in requirements traceability over the last decade. The thesis has addressed all the research questions defined to achieve aims and objectives. It has already been mentioned that the number of publications on requirements traceability started to rise after year 2000 (see Figure 6 Section 2.2.2). Therefore this effort of presenting a systematic review on requirements traceability can be treated as timely support for research community. The results of this systematic review could be of interest to research community as they help of identify gaps in current research within requirements traceability.

Based on the systematic review results it can be concluded that Gotel & Finkelstein provided comprehensive and state-of-the-art definition of requirements traceability. They define requirements traceability as the ability to describe and follow the life of requirements in both forward and backward direction throughout the system specification, development, deployment and refinement phases. It is to be noted here that tracing requirements in forwarding direction relates to the post-RS traceability while tracing requirements in backward direction relates to the pre-RS traceability. Therefore it is evident that for complete traceability these two aspects of traceability are essential.

The issues/challenges related to requirements traceability falls into three types. These are issues addressed by academia, issues addressed by industry and unsolved issues. Among these, unsolved issues are of interest and research community is working to solve these issues. Some of the prominent issues under this category are How much traceability is enough?, What is the right kind of traceability for a particular project?.

Similarly the requirements traceability tools can be classified into three types based on systematic review. These three types are requirements management tools providing traceability support, requirements traceability tools and other requirements management/traceability tools. Rational Requisite Pro, DOORS, Design Track and Scenario Advisor Tool are examples of requirements management tools providing traceability support. RETRO and TRAM are examples of requirements traceability tools. System Level Automation Tool for Engineers (SLATE), CRADLE and RDD-100 are examples of other requirements management/traceability tools.

The authors have classified requirements traceability techniques into two major types based on systematic review results. These are techniques supporting pre-RS traceability and techniques supporting post-RS traceability. It is interesting to know that there is only a single technique used for pre-RS traceability. This indicates that there is very less research work done in this area. So research community should focus on this area to improve pre-RS traceability. Again the techniques supporting post-RS traceability are sub-divided into three types. These three types are techniques favouring traceability of functional requirements, techniques favouring traceability of non-functional requirements and techniques favouring traceability of both functional & non-functional requirements. VBRT and FORT are examples of techniques favouring tracing of functional requirements. Design pattern, keywords & Ontology and goal centric traceability are examples of techniques favouring traceability of non-functional requirements.
requirements. Event-based traceability, IR methods and rule based approach are some examples of the techniques favouring traceability of functional and non-functional requirements.

This thesis also highlights the practice of requirements traceability in industry. For this purpose two companies have been interviewed. After performing analysis of industrial interviews, the authors have classified these two companies into high-end traceability users and low-end traceability users.

Based on the systematic review results, analysis of industrial interviews and after negotiation with interviewees five issues has been deduced (see Section 4.3 Table 22). These issues are solved by the proposed framework.

This framework is based on three levels. The level 0 handles pre-RS traceability while level 1 and level 2 deals with the post-RS traceability. Level 1 provides traceability for the functional requirements, where as level 2 provide traceability for non-functional requirements. The limitations and issues solved by each level are explicitly mentioned in the framework. Alternative solutions are also provided to overcome the limitations of each level.

TLFRT was statically validated in two companies. The main reason for validating this framework in industry was to get feedback form professionals working with requirements traceability and requirements engineering. The important points deduced form the industrial validations of TLFRT are given in Section 6.1.4.

Based on the validation feedback it is agreed that TLFRT is a customizable framework. The three levels of TLFRT are not compulsory and they may not be followed in a strict sequence. A company may use level 1 or level 2 to manage traceability without using level 0. The level 0 of this framework helps to trace requirements between requirements specification and their origin.

The level 1 of TLFRT prioritizes requirements based on three parameters which are value, risk and effort. This prioritization mechanism is responsible for reducing cost and efforts related to requirements traceability. In level 1 product manager or project manager can prioritize the requirements based on these three parameters. Where as stakeholders can only prioritize requirements based on value. During validation of TLFRT in company B the interviewee suggested that the person who owns the budget should prioritize the requirements based on these three parameters.

The level 2 of TLFRT takes care of NFR. In both the companies the interviewees agreed that the `impact analysis report` produced during level 2 is of importance for project managers and other stakeholders. Both the companies showed interest in using this framework by implementing it in the form of a tool. TLFRT uses RTM for maintaining traceability links. But according to systematic review results RTM are subject to scalability constraints. In this case alternate methods of dynamic link generation like IR methods or keywords and ontology can be used.

TLFRT uses three techniques to manage traceability at each level. Therefore when it is implemented in the form of a tool proper training should be given to use this framework. The interviewees form both the companies pointed out that the initial cost of TLFRT may be higher. But in general it is affordable for both high-end and low-end traceability users.

8.2 Future work

In this section, a number of areas that are regarded as important to study further are presented. This discussion points out the most important areas where research should be focused, based on the results of this thesis.

This framework has been statically validated in two companies developing software applications for telecommunications. Therefore thorough static validation of this framework should be done in more than two companies. In addition these companies should preferably be working in different domains.

One of the directions for future work primarily includes developing this framework in the form of a tool. This tool should be customizable so that companies may use level 1, level 2, level 3 or may be all levels of TLFRT depending on their needs.
When this framework is implemented in the form of a tool, then its dynamic validation should be carried out in two or more companies. This will help to evaluate the working of this framework in real industrial environment. It will also provide suggestions for improvements based on actual use of this framework in industry.

As a future work this framework can also be implemented in the form of a simulator. The internal working of level 1, level 2 and level 3 could be displayed with the help of simulations using MATLAB. This will help to present this framework to companies in much better way.

There is very less work done in the area of pre-requirements specification traceability. The research community is still working to solve challenges of traceability between requirements and their source needs. The unsolved issues related to requirements traceability (see Section 3.2 Table 9) like how much traceability is enough and what is the right kind of traceability for a particular project, could also be explored. The research community is still grappling with these unsolved issues.
REFERENCES


Last Visited on 2007-09-21


Last visited on 2007-09-25


APPENDIX 1: LIST OF REJECTED ARTICLES


APPENDIX 2: QUESTIONNAIRE FOR COMPANY ‘A’

Warm-up Questions

1. What is your name and designation?
   1.1. Person B: Requirements Manager
   1.2. Person A: Discipline Driver for PDU (product development unit)

2. What is your company’s name?
   Company A Karlskrona Sweden

3. What educational background do you have?
   3.1. Person B: University Graduate
   3.2. Person A: Upper Secondary school

4. What is your current job description?
   4.1. Person B: I am working as requirements manager in Company A. I am responsible for requirements engineering activities.
   4.2. Person A: I am working as Driver for PDU (product development unit) in Company A. I am responsible for managing traceability in our tool for requirements management. Previously I was working in graphics industry.

5. How long have you worked on the current position?
   Person B: Nine years
   Person A: Ten years

6. Since how long you are working in the process area of requirements traceability?
   Person B: 1.5 years
   Person A: 5 years

Brief Company Description

7. What are your customer’s type (Bespoke or Market-Driven)?
   Market-driven. We develop applications for telecomm operators.

8. Is your company ISO9000 or CMMI certified?
   ISO 9000 certified – No idea about CMMI level.

9. How many people in your organization are working in the process area of requirements traceability?
   There are thirty people working in the process area of requirements engineering but 3-5 people are directly working with requirements traceability.
Requirements Traceability

10. How requirements traceability is implemented in this organization?
   We use tool for requirements handling MARS developed by IBM particularly for Company A. There is a module in MARS that handles traceability.

11. What are the benefits of implementing requirements traceability?
   The main benefit is customer satisfaction. The customer can see every thing he asked for is implemented.
   Requirements traceability provides guarantee to requirements engineers and product managers that every requirement is implemented.
   Requirements traceability helps to ensure impact analysis and derivation analysis.

12. What attributes of requirements do you consider while implementing requirements traceability?
   Three attributes of requirements namely ID, slogan and description are of importance for requirements traceability in MARS.

13. How requirements traceability helps in change management (e.g impact analysis, derivation analysis, coverage analysis)?
   In our practice requirements traceability facilitates impact analysis. When a change request (CR) is initiated at any stage may be after project life cycle we can trace other requirements or design artifacts or test cases that can be affected.
   Similarly in coverage analysis requirements traceability ensures implementation of all requirements.

Problem Specific Questions

14. What kinds of issues/problems do you face during implementation of requirements traceability practices?
   The traceability is very well implemented in our tool. But one problem is the manual decomposition of master requirements specification (MRS) into detailed requirements for requirements traceability.
   Secondly when detail requirements are written it is difficult to find input requirements that matches them.
   The maintenance cost of MARS is also very high so we are thinking to shift to some other tool.

15. Describe the problems that normally occur?
   The change request (CR) is another problem. In company ‘A’ we normally have about 200 change requests in a project.

16. What are the factors responsible for these problems?
   These problems are related to requirements management. Requirements traceability also comes under requirements management.

17. How do you handle these problems?
   We have a traceability team of 3-5 people who are responsible for handling traceability issues. They can also get help for our main team working in requirements engineering. If we cannot
handle problem of requirements traceability as supported by MARS we can get technical help form IBM for traceability using MARS tool.

18. Does traceability costs more than it delivers?
   No not is our case. Traceability helps us to ensure that every requirements of sponsor is implemented. We can trace requirements to test case and it helps us in acceptance testing.

Tools and Techniques

19. What requirements traceability techniques you are using?
   We are not using any particular techniques like requirements traceability matrix. We use MARS for handling traceability issues.

20. Are you using some automated tools for implementing requirements traceability?
   Yes, company ‘A’ is using a module built in MARS for requirements traceability.

Summing Up Questions

21. Does your organization have developed its own standard or framework for requirements traceability?
   IBM has developed a tool MARS for requirements management for company ‘A’. We are using it for requirements traceability.
   In future we are thinking to move to another tool REQUISITE PRO for requirements traceability as maintenance cost of MARS is comparatively high.

22. Does your organization is doing R&D in requirements traceability?
   We do not have a specific department for R &D in requirements traceability.
APPENDIX 3: QUESTIONNAIRE FOR COMPANY ‘B’

Warm-up Questions

1. What is your name and designation?
   Person C
   Software Quality Engineer

2. What is your company’s name?
   Company B Ronneby Sweden

3. What educational background do you have?
   MSc (Software Engineering) from BTH.
   Licenciate degree (Software Engineering and process improvement) from BTH.

4. What is your current job description?
   My job title is Software Quality Engineer and my job description includes process development,
   matrix measurement, controlling the development of projects.

5. How long have you worked on the current position?
   I am working on my current position for two years.

6. Since how long you are working in the process area of requirements traceability?
   I am not directly working with requirements engineering but I have the knowledge of how
   requirements are treated, tested and verified in company ‘B’ and I also have the academic input
   along with some viewpoint of how it is applied in practice in industry. I am familiar with the
   process of requirements engineering since I am working as Software Quality Engineer in company
   ‘B’.

Brief Company Description

7. What are your customer’s type (Bespoke or Market-Driven)?
   We develop both bespoke and market-driven products for our customers. We are developing
   application suite and framework for development based on Symbian operating system and our
   customers are phone manufacturers. We do not develop the mobile phones but we develop the
   application suite for the mobile phones.

8. Is your company ISO9000 or CMMI certified?
   The company in which I am working is neither ISO9000 certified nor CMMI certified. There is no
   formal certification.

9. How many people in your organization are working in the process area of requirements
   traceability?
   In requirements gathering or capture part there are fifteen persons. Then in treatment of
   requirements which is a part of software development organization in our company there are
   thirty people. The role of software development organization in our company is to clarify
requirements, to process requirements and to trace them down to implementation and testing. It also includes the people from requirements gathering part.

Requirements Traceability

10. How requirements traceability is implemented in this organization?
The high level requirements we call features or feature specification are broken down into PREQ or product requirements which are a final detail. The product requirements are broken down into functional specification which is basically the lowest detail level. The traceability is maintained in a way that every product requirement has a connection with feature and vice versa. But the requirements traceability is not maintained with design, source code. In testing each functional specification is connected to at least one test case and that test case must have passed. This connection is not through design and code but it goes from requirements to testing phase. In this way traceability is implemented in our company.
The tool we are using is RMS (requirements management system) and it provides requirements traceability support but only within the requirements part.

11. What are the benefits of implementing requirements traceability?
We can get a better level of detail by traceability by breaking requirements into features and feature to functional specification. We can see the connection where the functional specification comes from because when we process the requirements at the high level or feature level, they are rather visionary or at abstract level. We can go back and forth by their parent child relationship. Traceability form feature specification to test case part ensures that the feature is complete. But in our company the linkages between feature specifications and test cases are kept manually. We have TMS (test management system) for managing and storing test cases with an ID. But there is no linkage between RMS (requirements management system) and TMS (test management system).

12. What attributes of requirements do you consider while implementing requirements traceability?
We attach ID with feature specification that we manually insert in RMS (Requirements Management System) and other important attributes for requirements traceability are feature description, dependencies with operating system, feature specification estimates. Estimates mean the time that a feature will take.

13. How requirements traceability helps in change management (e.g. impact analysis, derivation analysis, coverage analysis)?
The impact analysis can be done in several levels. If it is early in the development cycle in the project specification part before something is developed, it can be done by product manager who has sufficient competence to say that this change will affect the product in some way. But if development is on going the change request is done by the team that are managing that component affected by the change. This team may consist of software engineers and architects. This impact analysis is facilitated by requirements traceability.

Problem Specific Questions

14. What kinds of issues/problems do you face during implementation of requirements traceability practices?
We have a rather good traceability between feature specifications, product requirements and functional specification. But if you like to see the requirements completeness in terms of what are the functional specifications that have been developed and verified for specific project that is very
hard. There is no relation between project and functional specification in our tool. So we have to manually maintain the features that are verified for a particular project using MS excel sheets.

15. Describe the problems that normally occur?
   The problems are due to the requirements which are addressed by change request. A change request suddenly leads to a change in feature. There are normally 5-8 change requests for every twenty features. The change request could be very small, simple to do and they can also be very large.

16. What are the factors responsible for these problems?
   One factor naturally is the tool support. A good tool would help in keeping the traceability visible. You cannot invest in the tool but it is also the business case to see what would be benefit from good requirements trace.

17. How do you handle these problems?
   These problems are handled by manual effort and lot of personal responsibility. The project managers are keeping traceability logs in their heads, mails and it’s a high risk.

18. Does traceability costs more than it delivers?
   It’s hard to say. The traceability we have today in our company is rather cheap it does not cost any thing to maintain. The traceability from functional specification to test case may cost and I think that’s value for money. Our tool does not support traceability form functional specification to test case.

Tools and Techniques

19. What requirements traceability techniques you are using?
   We use requirements traceability matrix (RTM) using excel sheet for maintaining manual traceability from functional specification to testcases. The traceability between requirements is handled by our tool RMS (Requirements Management System).

20. Are you using some automated tools for implementing requirements traceability?
   Yes we are using an automated tool requirements management system (RMS) that provides support for traceability but only with in the requirements.

Summing Up Questions

21. Does your organization have developed its own standard or framework for requirements traceability?
   No we have not developed any standard or framework for requirements traceability. However a framework on requirements traceability could be of interest to our company.

22. Does your organization is doing R&D in requirements traceability?
   Not much. I think we would be benefited from looking at what we could gain from having traceability from requirements to design phase and down to source code. My personal belief is that we should look at how it can be done.
## APPENDIX 4: FRAMEWORK VALIDATION QUESTIONNAIRE FOR COMPANY ‘A’

### Level 0

1. Do you think that TLFRT facilitates in tracing requirements back to their sources?
   
   Yes TLFRT facilitates tracing requirements back to their sources at level 0 and we think that after level 0 requirements are complete & consistent due the directives with their optimized & finalized form.

2. Do you think that TLFRT helps to manage change request or requirements change?
   
   Yes we think that by using TLFRT it is easy to manage the change request, but it can be a bottleneck if there are 2500 changes requests out of 10000 requirement per year.

### Level 1

3. Do you think that tracing prioritized requirements using TLFRT helps to reduce cost and effort?
   
   Yes by prioritizing requirements cost and effort can be reduced but the prioritizing parameters should not be changed during the whole project.

4. TLFRT uses three prospective to prioritize requirements. Do you think that these prospective are enough for prioritizing requirements?
   
   Yes these 3 parameters are enough to assign priority to requirements.

5. In TLFRT only product manager or project manager can prioritize requirements based on three prospective cost, value and effort. Where stakeholders can only prioritize requirements based on one prospective which is value. Do you think that this kind of division is appropriate?
   
   Yes we think that this kind of division is reasonably good.

6. In the Level 1 of TLFRT the ‘requirements packaging’ step is optional. What is your opinion that this step should be optional or mandatory?
   
   ???????

7. TLFRT uses RTM for maintaining traceability links. Do you think that it is scalable to use RTM in projects of your company?
   
   We don’t have manual RTM. We are using requirement management tool which is responsible for traceability, so for our company this question is not valid.

8. Do you think that Level 1 of TLFRT helps to trace functional requirements?
   
   Yes Level 1 helps to trace functional requirements in both forward & backward direction.
<table>
<thead>
<tr>
<th>Level 2</th>
</tr>
</thead>
</table>
| 9 | In Level 2 of TLFRT Softgoal Interdependency Graph (SIG) is used to model NFR. Do you think that SIG is helpful to model NFR?  
*Normally in our company we treat both functional & non-functional requirements in the same way, but by using SIG may be initially it is difficult to model NFR but when you learn it, it will be very easy to model the NFR using SIG.* |
| 10 | In Level 2 of TLFRT traceability links are established between set of elements in SIG affected by the functional change and functional model. This functional model can be design documents or code. In your opinion due to functional change in any NFR the traceability links should be established between design documents / code or both of them?  
*??????* |
| 11 | The output of Level 2 of TLFRT is an impact analysis report. Stakeholders can use this report to decide whether to implement proposed change or discard it. Do you think that this ‘impact analysis report’ is beneficial for making decisions?  
*Yes of course* |
| 12 | Do you think that Level 2 of TLFRT helps to trace NFR?  
*Yes TLFRT provide easy way to trace non-functional requirements in addition SIG provide a clear picture for doing tradeoff between non-functional requirements.* |

<table>
<thead>
<tr>
<th>General</th>
</tr>
</thead>
</table>
| 13 | Do you think it is easy to use this framework?  
*Yes* |
| 14 | Do you think it is easy to customize this framework according to your needs?  
*Yes of course TLFRT is customizable framework plus you provided some alternative solutions for different number of requirement and for different nature of requirements.* |
| 15 | Do you think this framework is affordable for both low-end and high-end traceability users?  
*Yes we think that TLFRT is affordable for both low-end and high-end traceability users. Initially it can take more cost but once it is developed then it will be quite economical.* |
| 16 | Do you think there is any overhead in the implementation of this framework?  
*No we don’t think there are any overheads in the implementation of TLFRT.* |
| 17 | Can you name and explain most important potential shortcomings with the framework as you observe it?  
*Right now we are unable to answer this question— we will email you after a thorough study of TLFRT.* |
| 18 | Can you name and explain most important benefits with the framework as you observe it?  
*Right now we are unable to answer this question- we will email you after a through study of TLFRT.* |
| 19 | Would you like to suggest some changes/enhancements in this framework?  
*Right now we are unable to answer this question- we will email you after a through study of TLFRT.* |
## APPENDIX 5: FRAMEWORK VALIDATION QUESTIONNAIRE FOR COMPANY ‘B’

<table>
<thead>
<tr>
<th>Level 0</th>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Do you think that TLFRT facilitates in tracing requirements back to their sources?</td>
<td>Yes TLFRT helps in tracing requirements back to their sources. This is very important to identify who was the issuer of a particular requirement or to identify ambiguous or incomplete requirements.</td>
</tr>
<tr>
<td>2</td>
<td>Do you think that TLFRT helps to manage change request or requirements change?</td>
<td>Yes I think it can manage change requests.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Level 1</th>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Do you think that tracing prioritized requirements using TLFRT helps to reduce cost and effort?</td>
<td>Yes I think that we can reduce the cost and effort by prioritizing requirements. As you also have empirical proof of this from your systematic review, so I think this is a practically valid claim.</td>
</tr>
<tr>
<td>4</td>
<td>TLFRT uses three prospective to prioritize requirements. Do you think that these prospective are enough for prioritizing requirements?</td>
<td>Yes I think that these three parameters which are risk, value and effort are enough for prioritization.</td>
</tr>
</tbody>
</table>
| 5       | In TLFRT only product manager or project manager can prioritize requirements based on three prospective cost, value and effort. Where stakeholders can only prioritize requirements based on one prospective which is value. Do you think that this kind of division is appropriate? | I do not agree with this kind of classification of control to prioritize requirements between product manager/project manager or stakeholders. I think it depends upon who owns the budget. This kind of classification is good when project manager or product manager owns the budget. If you only prioritize on value then you do not see risk and effort associated with it.  
If there is a project office in your company that owns the budget and it is composed of developers. Then they need to know about risk and effort in addition to value for prioritizing the requirements.  
In our company we have a separate department called ‘project management’ that owns the budget. It does not contain any project manager/product manager and they decide what features should be added in the product.  
But I see that you think project manager or product manager as budget owner as well and... |
he has control of all resources. In this case this classification of control is ok.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
| 6 | In the Level 1 of TLFRT the ‘requirements packaging’ step is optional. What is your opinion that this step should be optional or mandatory?  
   | Well I do not see this step to be considered as mandatory. If the technique you are using for Level 1 has ‘requirements packaging’ as optional, it is better to stick to it as optional. |
| 7 | TLFRT uses RTM for maintaining traceability links. Do you think that it is scalable to use RTM in projects of your company?  
   | Yes we are using RTM in our daily practice and we think that they are easy to use and perhaps the cheapest solution for managing traceability. You are mentioning some other methods to take care of potential problems with RTM like scalability. This is a very good approach to provide alternate solutions. But for our company RTM are good for manual traceability. |
| 8 | Do you think that Level 1 of TLFRT helps to trace functional requirements?  
   | Ye I think that Level 1 takes care of functional requirements. |

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
| 9 | In Level 2 of TLFRT Softgoal Interdependency Graph (SIG) is used to model NFR. Do you think that SIG is helpful to model NFR?  
   | I think that SIG is helpful to model NFR. But the only problem is that you need to be familiar with SIG and its notations to use it for modeling NFR. In our company we do not use SIG and we do not model NFR. |
| 10 | In Level 2 of TLFRT traceability links are established between set of elements in SIG affected by the functional change and functional model. This functional model can be design documents or code. In your opinion due to functional change in any NFR the traceability links should be established between design documents / code or both of them?  
   | Well I think its good to do so if it is also used in the technique you are using at Level 2 to handle traceability for NFR. |
| 11 | The output of Level 2 of TLFRT is an impact analysis report. Stakeholders can use this report to decide whether to implement proposed change or discard it. Do you think that this ‘impact analysis report’ is beneficial for making decisions?  
   | Yes I think this report is good for project managers and other stakeholders. |
| 12 | Do you think that Level 2 of TLFRT helps to trace NFR?  
   | Yes I do think that Level 2 helps to trace NFR. |

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
| 13 | Do you think it is easy to use this framework?  
<p>| There are mainly three techniques that you need to be aware of inorder to use this... |</p>
<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do you think it is easy to customize this framework according to your needs?</td>
<td>Yes we can customize TLFRT according to our company needs. As TLFRT is based on three levels. So we can do Level 1 and for this we do not necessarily need to do Level 0 first ( if I have understood correctly). When we trace functional requirements then if we need to know where they are coming from we can do Level 0.</td>
</tr>
<tr>
<td>Do you think this framework is affordable for both low-end and high-end traceability users?</td>
<td>Yes I think that TLFRT is affordable for both low-end and high-end traceability users. But I think that its initial cost will be higher and we also have to provide training to our employees to use TLFRT.</td>
</tr>
<tr>
<td>Do you think there is any overhead in the implementation of this framework?</td>
<td>No there are no overheads in the implementation of TLFRT. I think this framework is customizable. However as I have said before its initial cost may be higher.</td>
</tr>
<tr>
<td>Can you name and explain most important potential shortcomings with the framework as you observe it?</td>
<td>I do not see any shortcomings. If this framework is developed in the form of an automated tool using some high level language. Then it will be easy to use as there will be menus and buttons for moving between various levels and taking care of some technical methods like modeling NFR in SIG. However still you need to train your employees.</td>
</tr>
<tr>
<td>Can you name and explain most important benefits with the framework as you observe it?</td>
<td>It is very good that you use RAM model in Level 0. It is practically true that requirements many not come form users only. Requirements may come form other high level or low level requirements. So in this case RAM can handle them. The other good feature is that you are explicitly mentioning the limitations of each level in TLFRT. I agree that although RTM are subject to scalability constraints but it is easy to use RTM and we also use them in our practice. It is really good that you are providing alternatives to RTM like IR methods for automatic link creation. But in order to keep this framework simple I think it is better to use RTM. If we go for a cheaper solution and low investment in beginning, then people will start using this framework. Later on in advance version of TLFRT you can use IR methods.</td>
</tr>
<tr>
<td>Would you like to suggest some changes/enhancements in this framework?</td>
<td>I think that the framework is well formulated and covers the most parts that are important.</td>
</tr>
</tbody>
</table>