Requirements Traceability Approaches: A Systematic Literature Review

Nejood Hashim Al-walidi
Faculty of Graduate Studies for Statistical Research,
Cairo University, Cairo, Egypt
email address:nejo.Hashim@su.edu.ye
ORCID : 0000-0002-1774-0801

Nagy Ramadan Darwish
Faculty of Graduate Studies for Statistical Research,
Cairo University, Cairo, Egypt
email address: n.nagy@fci-cu.edu.eg

Ali Hussein Zolait
College of Information Technology
University of Bahrain
Manama, Bahrain
email address: azolait@uob.edu.bh
ORCID : 0000-0002-8020-8084

***Abstract*—** Requirements traceability (RT) is a significant quality factor in software development, enabling software engineers to track requirements from inception to fulfillment. While previous studies have predominantly focused on singular aspects of requirements traceability, our Systematic Literature Review (SLR) delves into multiple often-overlooked facets. Our primary focus is on RT approaches, acknowledging a significant gap in research attention in this area. The objective of this research is to comprehensively explore requirements traceability approaches, their empirical evidence, and associated challenges. By doing so, we aim to lay a foundation for future research endeavors in this domain. Additionally, we seek to examine the latest real-time RT approaches, the criteria utilized for their evaluation, and the distinguishing characteristics of the identified methods. Adhering to SLR guidelines, we meticulously analyze, evaluate, and interpret relevant primary research spanning from 2006 to 2019. Our systematic literature review (SLR) identifies state-of-the-art approaches in requirements traceability, highlights gaps for further investigation, delineates criteria for evaluating traceability approaches, and outlines key characteristics of identified methods. This compilation serves as a valuable resource for both researchers and practitioners seeking specific RT approaches tailored to their interests. While prior studies typically focused on singular topics related to requirements traceability, our SLR casts a wider net, exploring numerous neglected dimensions of this critical aspect of software development. Our analysis specifically targets the period between 2010 and 2019.

Keywords*— *Requirements traceability, traceability approaches, Software development, Requirements traceability categories and challenges, Systematic Literature Review, evaluation

I. INTRODUCTION (*HEADING 1*)

In response to the high rate of software project failures, many software development standards have been proposed. These standards include SEI's CMMI and IEEE's JSTD-016. A common feature of these standards is that they all impose requirements traceability practices on the software development process [1]. Requirements traceability can be defined as "*the ability to describe and follow the life of a requirement, in both forward and backward direction, (i.e., from its origin, through its development and specification, to its subsequent deployment and use, and through periods of ongoing refinement and iteration in any of these phases)*" [3]. This definition can be visualized in Figure 1.

As shown in Figure 1, requirements traceability ensures continued alignment between stakeholder requirements and the various outputs of the system development process. Consequently, requirements traceability has been demonstrated to provide many benefits in software development. Requirements traceability can demonstrate that a system meets its specified requirements. Additionally, it simplifies identifying which requirements, design elements, code, and test cases need updates to accommodate a change request during the software project's maintenance phase. Moreover, by following traceability links, a project manager can promptly determine the number of artifacts impacted by a proposed change, allowing for informed decision-making regarding the associated costs and risks.

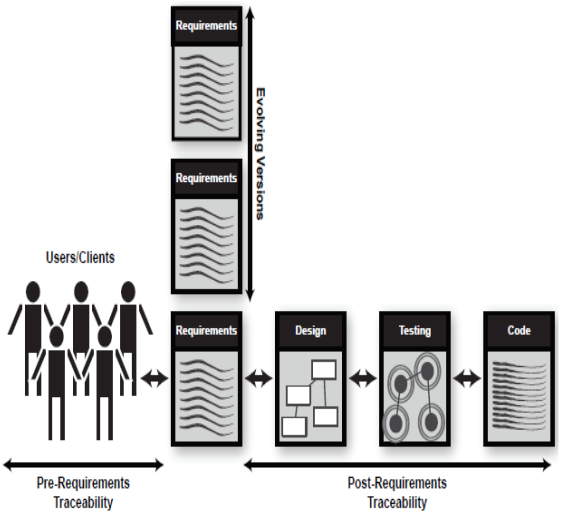

Fig 1. A View of Software Requirements Traceability [1]

Although the importance of traceability seems to be generally approved in the software engineering industry, organizations continue to struggle to implement it. One major challenge facing the implementation of traceability is simply the costs involved. Therefore, it is important to address questions such as "How much traceability is enough?" and "What kinds of traceability provide cost-effective solutions?" [2].

In previous Systematic Literature Reviews (SLRs) within the realm of Requirements Traceability (RT), researchers have primarily focused on investigating related challenges, as well as the approaches and tools developed to tackle them. The insights gained from these SLRs can serve as valuable resources for both researchers and practitioners seeking specific sets of approaches and tools tailored to their interests. However, these SLRs have varied in their scopes: some have delved into RT definitions, challenges, tools, and techniques using primary studies

spanning the years 1997-2007 [3]; others have focused on recording and maintaining information of the traceability within the context of Model-Driven Engineering [4]; while some have paid particular attention to traceability between the architecture of the software and code, presenting classification schemes to distinguish various aspects of traceability approaches [5]. Additionally, there have been SLRs conducted to explore the latest developments in the area of requirements traceability, utilizing primary studies from the years 2010-2017 [6]. To our knowledge, there is currently no recent SLR available that analyzes and evaluates the latest RT approaches. During the period between 2010 and 2019, the majority of researchers concentrated on specific topics related to RT approaches as mentioned above. Consequently, our objective is to explore the areas that have been overlooked within the domain of RT approaches during this time frame. The main objective of this research is to investigate current RT approaches, focusing on their evaluation against specified criteria, and to offer a framework for situating new research endeavors appropriately. We aim to address aspects that may not have been adequately covered in previous studies.

To achieve this objective, we follow the SLR guidelines outlined by "Kitchenham and Charters" [7], conducting a comprehensive analysis, evaluation, and summary of relevant primary research conducted between 2010 and 2019.

The rest of this systematic literature review (SLR) is organized as follows: Section 2 reviews related literature. Section 3 outlines the methodology used for this SLR. Section 4 presents the results obtained from data analysis. Section 5 discusses these results in the context of the research questions. Finally, Section 6 offers our conclusions and suggests directions for future work.

II. RELATED SLRS

In their paper titled "*Requirements traceability: A systematic review and industry case study*", the authors investigated four research questions "*What is requirements traceability based on state-of-the-art research? What are the challenges when implementing requirements traceability and how does research address these challenges? What are the various requirements for traceability tools according to research literature? What requirements traceability techniques are covered in the research literature?*". The results indicate that the authors provided several common definitions, challenges, tools, and techniques. The most commonly used definition of RT is narrated as "the ability to describe and follow the life of a requirement in both forwards and backward direction (i.e., from its origins, through its development and specification to its, subsequent deployment and use, and periods of on-going refinement and iteration in any of these phases)", was the most commonly used definition (about 80%). An interesting observation is that most frequently serves as the primary reason for not implementing and maintaining adequate traceability policies. The requirements traceability tools covered in their SLR include requirements tracing on-target (RETRO), Rational RequisitePro, DOORS, DesignTrack, TRAM, or tool for requirements and architectural management, Scenario Advisor, and other traceability tools like SLATE, CRADLE, RDD-100, Marconi RTM, RTS, Rtrace, and Teamwork/RQT. The requirements traceability techniques covered in the SLR include Value-Based Requirements Tracing(VBRT), Feature-Oriented Requirements Traceability(FORT), Pre-RS requirements traceability, Event-Based Traceability(EBT), Information Retrieval(IR), Rule-Based Approach(RBA), Feature-Model based approach, Scenario-based approach, process centered engineering environments, Design Patterns, traceability matrices, keywords and ontology, aspect weaving, Goal-Centric Traceability, and hypertext-based methods: Most of these techniques and tools were not validated empirically[3].

The paper titled "*Model-Driven Engineering as a new landscape for traceability management*". The authors investigated five RQs about "To what extent do methodological proposals recommend automating the generation of trace links? How do these proposals suggest managing and analyzing traceability? Are there tools or frameworks that offer technological support for traceability management in Model-Driven Engineering (MDE)? What are the current limitations in traceability management within the context of MDE? Are there specialized journals or conferences that focus on traceability management in MDE?". The findings suggest that out of a total of 10,028 results, only 267 were considered relevant studies. After removing duplicates, 157 unique studies were assessed against predetermined exclusion criteria, resulting in 29 primary studies. These primary studies were further grouped into 17 Groups of Primary Studies for subsequent review stages. Overall, the authors noted satisfactory research quality across all evaluated proposals, with each achieving a minimum quality score of 50%. Notably, GPS4 emerged as the top proposal based on quality assessment. The primary studies covered various topics, including trace generation (automated and/or manual), metamodel (general or specific purpose), trace management (storage, visualization, supported operations, and analysis), and implementation (complete toolkit or partial). In terms of quality assessment, the absence of clearly defined research methods was identified as a significant limitation. The authors expressed interest in investigating any potential correlation between the quality of studies and their publication venue. Additionally, they emphasized their readiness to contribute to the development or refinement of methodological and technical proposals for addressing traceability in Model-Driven Engineering (MDE) [4].

In their paper entitled "*A systematic literature review of traceability approaches between software architecture and Source Code*". The researchers investigated six RQs that seek clarify for inquiries regarding what is the current state of traceability methods and tools between software architecture and the source code. Specifically, what information is available for tracing from higher-level architectural artifacts to lower-level artifacts such as source code and vice versa? Additionally, what empirical evidence

has been documented in the field of traceability between software architecture and source code? As well as "to what extent are the reported traceability relationships useful in understanding software architecture? What are the reported benefits and liabilities of traceability approaches between software architecture and the source code? What are the reported issues, barriers, and challenges of traceability between software architecture and the source code?". The findings indicate that the authors have pinpointed the latest advancements in requirements traceability methods and tools, which connect software architecture to its source code. Additionally, they've underscored areas where improvements are needed and proposed avenues for future study. The systematic literature review (SLR) delves into requirements traceability methods, encompassing Event-Based Traceability, Rule-Based Traceability, Hypertext-Based Traceability, Traceability Based on Information Retrieval, Design Patterns Based Traceability, Model-driven Traceability, and Traceability Based on Machine Learning Techniques. The authors propose a classification framework to discern different facets of these approaches, focusing on their Nature, Automation, Types of relations, Granularity, and Direction and representation of traceability information. This classification offers a basis for researchers and practitioners to select or explore specific approaches [5].

In their paper entitled *"Requirement traceability techniques and tools",* the researchers investigated two research questions to address "What are the leading models, challenges, and tools in the area of requirement traceability for the period from 2010 to 2017? What are the pros and cons of leading requirement traceability models and tools?". The results show that the authors identified and investigated 33 research studies published during 2010-2017. The study identified 7 models, 10 challenges, and 14 tools in total. Among the models discussed were the Traceability Information Model (TIM), Traceability Meta Model, Traceability Process Model, Traceability Assessment Model (TAM), Semantic Model, I-Trace, and Requirements Dependency Model. The challenges highlighted encompassed issues such as traceability decay, lack of guidance, commitment issues, difficulties with manual traceability, gaps in knowledge and understanding, project longevity concerns, conflicting stakeholder perspectives, communication gaps between teams, human factors, biases, and inadequate tool support. Various tools were examined, including ECOLABOR, TOOR, RESAT, POIROT, CREWS-EVE, ProR, Trace Analyzer, TRIC, ADAMS, SCOTCH+, Trace Maintainer, DOORS, RequisitePro, and RETRO. The authors provided a thorough analysis of both models and tools, ultimately concluding that DOORS and the Traceability Meta Model stood out as the most effective requirement traceability tool and model, respectively. However, they noted that a comprehensive exploration of requirement traceability challenges was not achieved. The authors planned to conduct an in-depth examination of the identified challenges in their forthcoming article. [6].

III. Research Methodology

A Systematic Literature Review (SLR) functions as a structured method to locate, evaluate, and analyze all extant research relevant to a particular research question, field of study, or notable occurrence. The individual studies contributing to a systematic review are referred to as primary studies, making a systematic review a type of secondary study. The significance of SLR lies in its ability to synthesize existing information comprehensively and impartially, enabling the formulation of broader conclusions about a subject or serving as a precursor to further research endeavors [7]. This study aims to discern the prevailing methodologies employed for requirements traceability between 2006 and 2019, while also identifying areas necessitating further investigation. Adhering to the guidelines for Systematic Literature Review, we scrutinized, assessed, and interpreted the relevant studies available. A systematic literature review typically encompasses three principal phases:

- Planning the review phase (See Section 3.1)
- Conducting the review phase (See Section 3.2)
- Reporting the review phase (See Sections 4 and 5).

Planning the review phases involves three steps: Specifying RQs, developing a Search Strategy, and Defining Study Selection Criteria. **Conducting the review phase** consists of three steps: Selection of Primary Studies, Quality Assessment, and Data Extraction and Synthesis. **Reporting the review phase** comprises three steps: Presenting SLR Results, Discussing SLR Findings, and Concluding, as shown in Figure 2.

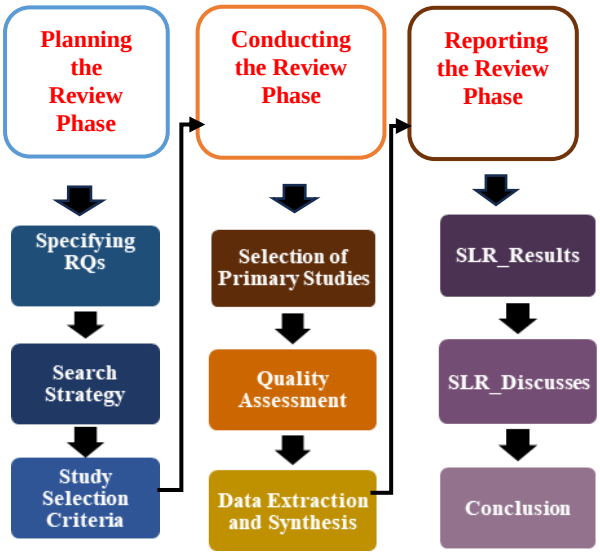

Fig 2. A Systematic Literature Review Phases

A. *Planning the review*

The objective of this phase is to formulate a review protocol, which involves defining the Research Questions (RQs) intended to be addressed by the review (Refer to Section 3.1.1), determining the search strategy to locate and identify primary studies (Refer to Section 3.1.2, and

establishing the criteria for selecting studies (Refer to Section 3.1.3).

- ***Specifying the Research Questions***

Identifying the research questions stands out as a pivotal aspect of any systematic literature review, given that these questions guide the entire methodology of the review [5] [7]. Presented below are the chosen research questions intended to fulfill the goals of our SLR:

RQ1: What are the challenges related to RT?
RQ2: What are the state-of-the-art RT approaches?
RQ3: What are the criteria used to evaluate RT approaches?
RQ4: What are the characteristics of the identified RT approaches??

- ***Search Strategy***

In this SLR, the search procedure involved conducting online searches using specific search terms and utilizing online resources. Table 1 delineates the digital libraries employed in the SLR to explore requirements traceability methodologies:

TABLE 1. THE ONLINE RESOURCES USED IN THE SLR

Name	Website	Name	Website
"IEEE Xplorer"	"https://ieeexplore.ieee.org"	Semantic scholar	"https://www.semanticscholar.org"
"ACM Digital Library"	"https://dl.acm.org"	World Scientific	"https://www.worldscientific.com"
"Google Scholar"	"https://scholar.google.com"	Research gate	"https://www.researchgate.net"
"Academia"	"https://www.academia.edu"	Springer	"https://www.springer.com"

The search terms below were employed to gather and extract primary studies covering the timeframe from 2006 to 2019:

- Requirements Traceability (RT)
- Requirements Tracing.
- Traceability
- Requirements Traceability Approaches
- Requirements Traceability Technique.
- (1) AND Issues
- (1) AND Techniques
- (1) AND Challenges
- (1) AND Direction
- (1) AND Types
- (2) AND Types
- (3) AND Issues
- (4) AND Evaluation
- (4) AND Criteria
- (4) AND Characteristics

- ***Study Selection Criteria***

The study of the selection criteria is aimed to identify primary studies that provide direct evidence about the research questions [7]. Inclusion and exclusion criteria are employed to assess the suitability of publications and decide whether to include or exclude specific studies from the SLR [5]. The inclusion and exclusion criteria used in this SLR are detailed in Table 2.

TABLE 2. STUDY SELECTION CRITERIA

Inclusion Criteria	Exclusion Criteria
• Do the studies during the 2006 and 2019- time span? • Primary studies that provide evidence about the RQs. • Are the collected citations relevant? • Journal articles and conference proceedings on traceability challenges, traceability approaches, traceability criteria, and traceability approach characteristics.	• Primary studies outside the period from the 2006 to 2019-time span. • Primary studies are not relevant to the research questions. • Studies not published in refereed journals or conferences. • Duplicate primary studies are included only once, using the latest version.

B. Conducting the Review Phase

Conducting the review involves three sub-sections: Selection of Primary Studies (See section 3.2.1), Study Quality Assessment (See section 3.2.2), and Data Extraction (See Section 3.2.3).

- ***Selection of Primary Studies***

In accordance with the research objective, a search string was formulated to identify primary studies based on the predefined inclusion and exclusion criteria within the timeframe of the systematic literature review (SLR) (See Section 3.1.3). Following the application of these criteria, 50 studies were selected from an extensive database comprising 250 studies, from which conclusions regarding requirements traceability approaches were drawn. Table 3 enumerates the 43 studies identified as primary studies.

- ***Study Quality Assessment (SQA)***

The SQA criteria are used to examine the accuracy and trustfulness of the used research methodology as well as the relevance of the citations [5]. The following criteria of quality have been applied:

- Title*: Is the title of research or keywords including the strings: ''traceability'', ''criteria'', ''techniques'', ''approaches'', ''characteristics'', and ''challenges''.*
- Abstract: *Does the abstract lead us to conclude that the main purpose of the study is requirements traceability?*
- Result*: Does the result relate to the requirements problem?*
- Conclusion: Were both negative and positive findings fully reported? Were there any limitations that influenced the conclusions or suggested avenues for future research?

Table 3. Primary Studies used in our SLR

Ref	Title
[8]	"A Model for Enhancing Requirements Traceability and Analysis"
[9]	"Grand Challenges, Benchmarks, and TraceLab: Developing Infrastructure for the Software Traceability Research Community"
[10]	"The Barriers to Traceability and their Potential"
[11]	"Tackling the term-mismatch problem in automated trace"
[12]	"A_MANY-TO-MANY_MN_RELATIONAL_MODEL_TOIMPROVE_REQU"
[13]	"A Review of Traceability Research at the requirements engineering"
[14]	"Addressing Traceability Challenges in the Development of Embedded Systems"
[15]	"ANewModelforRequirementstoCodeTraceabilitytoSupportCode CoverageAnalysis"
[16]	"Comparison of Information Retrieval Techniques for Traceability Link Recovery"
[17]	"Formulating a Software Traceability Model for Integrated"
[18]	"Gray Links in the Use of Requirements Traceability"
[19]	"Trust-based Requirements Traceability"
[20]	"Requirements Change Impact Analysis Using Event Based"
[21]	"Towards automated traceability maintenance"
[22]	"Utilizing Multifaceted Requirement Traceability Approach a Case Study"
[23]	"Effective and efficient requirement traceability the software development and information technology"
[24]	"AchievementsandChallengesinState-of-the-ArtSoftwareTraceabilityBetweenTest"
[25]	"RETRATOS Requirement Traceability Tool Support"
[26]	"Requirement Traceability for Software Development Lifecycle"
[27]	"Change Impact Analysis with a Goal-Driven"
[28]	"Rule-based Impact Analysis for Heterogeneous Software Artifacts"
[29]	"Test Coverage Measurement and Analysis on the Basis of software traceability approaches"
[30]	"Toward Multilevel Textual Requirements Traceability"
[31]	"Requirement Tracing using Term Extraction"
[32]	"Traceability StrategStudy"
[33]	"Comparative Study on Traceability Approaches in Software Development"
[34]	"The Grand Challenge of Traceability (v1.0)"
[35]	"Rule-Based Maintenance of Post-Requirements Traceability Relations"
[36]	"Using Semantics-Enabled Information Retrieval in Requirements Tracing"
[37]	"Survey on Usage Scenarios for Requirements"
[38]	"Improving IR-based traceability recovery"
[39]	"Goal-Centric TOSEM"
[40]	"Assessing traceability of software engineering artifacts"
[41]	"Can LSI help Reconstructing Requirements Traceability in Design and Test"
[42]	"From Frequency to Meaning: Vector space models of semantics"
[43]	"Normalizing source code vocabulary"
[44]	"A Traceability Metamodel for Change Management of Non-Functional"
[45]	"Enabling Collaboration in Distributed Requirements"
[46]	"Rule-Based Maintenance of Post-Requirements Traceability Relations"
[47]	"Enabling Automated Traceability Maintenance by Recognizing Development Activities Applied to Models"
[48]	"Tracing Non-Functional Requirements"
[49]	"A survey of traceability in requirements engineering and model-driven development"
[50]	"Decision-Centric Traceability of Architectural Concerns"
[51]	"Study of query expansion techniques and their application in the biomedical information retrieval"
[52]	"An integrated model for information retrieval based change impact analysis"
[53]	"An approach for integrating the prioritization of functional "and nonfunctional requirements"

- ***Data Extraction and Synthesis (DES)***

This step outlines how the necessary information from each primary study will be gathered and combined. Typically, researchers employ a form or table to collect data, facilitating the organization of the extraction and synthesis process. This approach enables researchers to address the research questions identified in section (3.1.1). Table 4. shows the elements of data extraction and synthesis for this SLR.

TABLE 4. ELEMENTS OF DATA EXTRACTION AND SYNTHESIS

#	Description	Details
1	Bibliographic Information	Authors Names, year of publication, the title of the publication, etc.
	Extraction of Data	
2	Overview	The chosen study aims to identify challenges associated with real-time (RT) systems, explore state-of-the-art RT approaches, examine the criteria used to evaluate these approaches and analyze the characteristics of the identified RT approaches.
3	Results	Results achieved in the chosen study.
	Synthesis of Data	
4	Challenges	The challenges related to RT (See Section 4)
5	Approaches	The state-of-the-art RT approaches (See Section 4)
6	Criteria	The criteria used to evaluate RT approaches (See Section 4)
7	Characteristics	The characteristics of the identified RT approaches (See Section 4)

C. Reporting the Review

Reporting the review includes two sections: *the systematic literature review results which include the analysis of the result obtained from this SLR (See Sections 4)* and *the systematic literature review discusses (See Sections 5)* which includes the answer to the fourth RQs, which was posed in (Section 3.1.1).

IV. THE SYSTEMATIC LITERATURE REVIEW RESULTS

In this section, we present the results of our SLR on requirements traceability approaches, which were conducted following the methodology described in Section 3 after analyzing the selected primary studies. We utilized 8 digital libraries to collect and identify relevant research (as detailed in Section 3.1.2). In total, 43 studies were analyzed to address the four research questions, as illustrated in Figure 3.

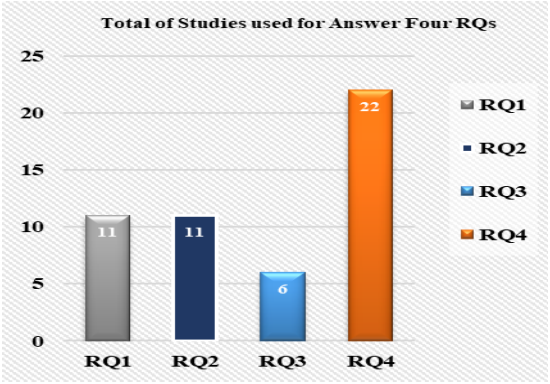

Fig 3. Total of Studies Used to Answer the Four RQs

The primary studies address the four research questions outlined in the results section. For RQ1, the papers referenced are (8, 9, 10, 11, 12, 13, 14, 15, 16, 18, 34). For

RQ2, the papers referenced are (17, 18, 19, 20, 21, 22, 25, 26, 27, 35, 39). For RQ3, the papers referenced are (24, 29, 27, 49). For RQ4, the papers referenced are (16, 17, 19, 21, 22, 23, 24, 29, 30, 31, 33, 36, 37, 38, 39, 40, 41, 42, 43, 44, 46, 47, 48, 49).

These primary studies were utilized to address four questions, each requiring a varying number of papers. Some of these papers were employed multiple times to tackle the research questions (RQs), as each paper alone was insufficient to comprehensively address a specific RQ. For instance, Paper 18 was utilized to address both RQ1 and RQ2. Additionally, Paper 27 was employed to answer RQ2 and RQ3, whereas Papers 24, 29, and 49 were utilized to address RQ3 and RQ4. Furthermore, Papers 17, 21, and 39 were utilized to answer both RQ2 and RQ4.

In addressing the first research question, we categorized the challenges into four main categories: Challenges in Technology, Challenges in Management, Social Challenges, and Technical Challenges, each with different types. We based our findings on 11 studies, although it's worth noting that some of these studies did not specify particular challenges.

Transitioning to the second research question, we've delineated five types of state-of-the-art approaches utilized for requirements traceability: Rule-based, Value-based, Information retrieval, Event-based, and Goal-centric. Our analysis was informed by 11 studies delving into these approaches.

For the third research question, we identified fifteen types of requirements traceability criteria based on insights from six relevant studies.

The criteria include database support [29], software traceability support, tool support, coverage construction [24], traceability scheme, scalability [49], visualization, change impact analysis, evaluation [27], trace type [24], trace direction [24], traceability analysis [24], underlying technique, and accessibility [24].

Finally, to address the last research question, we analyzed 20 studies to identify the primary characteristics of requirements traceability approaches. Further details can be found in Section 5).

V. The Systematic Literature Review Discusses

This section offers an analysis of the findings discussed in Section Four, aiming to address the four research questions outlined in Section IV. The results encompass challenges related to requirements traceability, state-of-the-art approaches, evaluation criteria for traceability approaches, and the key characteristics of identified requirements traceability approaches. Figure 4 illustrates the annual distribution of primary studies from 2006 to 2019. The figure indicates that the highest number of publications on requirements traceability approaches was achieved in 2012.

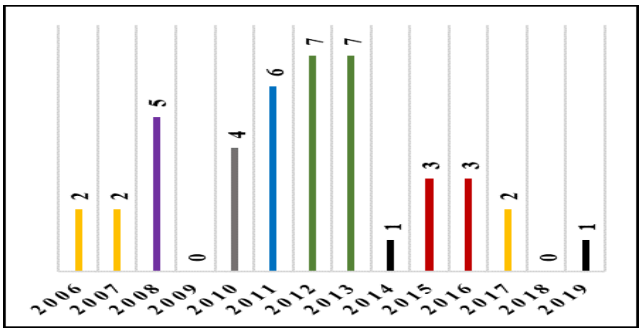

Fig 4. The Number of Papers used, Categorized by Years

1. *The Challenges of Requirements Traceability (Question1)*

The challenge of requirements traceability relates to the obstacles encountered when endeavoring to establish and maintain a comprehensive and coherent link among various project components, such as requirements, design, code, and testing. This connection, frequently referred to as traceability, plays a pivotal role in guaranteeing the correct implementation and validation of each requirement during the entirety of the project's life cycle. Requirements Traceability (RT) emerges as one of the most crucial and daunting tasks in ensuring the clarity and conciseness of requirements [8]. It holds a pivotal position in the software development process, with the potential to enhance the quality of software products in a positive manner [9]. According to Regan and his co-author [10], challenges in requirements traceability can be categorized into four types: Challenges in Technology, Challenges in Management, Social Challenges, and Technical Challenges. Figure 5 illustrates various types of challenges, with each category having distinct types. The 'Challenges in Technology' category includes four types, 'Challenges in Management' comprises four types, 'Social Challenges' involves six types, and 'Technical Challenges' encompasses six types. The following provides a description of these types.

These challenges have various effects on RT, demanding additional time and effort for resolution. At times, they result in project delays or even cancellations. Figure 5 shows the main challenges along with the total number of each type.

Fig 5. The challenges and their types in total

Figure 6 summarizes the most significant challenges in requirements traceability, categorized by type.

A. Technology Challenges

Technology Challenges encompass the following issues: traceability decay, low accuracy of traceability recovery methods, and inadequate presentation and visualization of trace links.

- ***Traceability decay:*** This type of problem occurs when trace links are not updated when any changes take place, resulting in a decline in traceability relations in which some trace connections are lost and others falsely reflect relationships. Links between two artifacts requirements artifacts and source code artifacts are particularly susceptible to this type of problem since developers regularly alter the source code without updating the links [10].

- ***Low accuracy of traceability recovery methods:*** The most commonly used techniques for establishing traceability links are Information Retrieval (IR) techniques. However, experiments across various domains and artifacts have demonstrated that these techniques typically exhibit low precision. This issue arises because IR-based methods connect artifact pairs based on textual similarity, which merely indicates the likelihood of a relationship between the two artifacts [11].

- ***Poor presentation and visualization of trace links:*** Large-scale undertakings are typically characterized by a large number of artifacts and a large number of trace links. Often, to represent the data, there are some tools, such as lists or mega tables, that are used. These types of tools hinder interested stakeholders from understanding the data on traceability and detecting inconsistencies [12].

- ***Lack of change notification and propagation:*** Every modification to one artifact affects all associated artifacts and links in the trace. In the worst-case scenario, for example, a change in the requirements will affect other artifacts that have relationships with it, such as design, source code, and test cases [13].

B. Management Challenges

Management Challenges include: *Cost, Obtaining Information, Organizational Problems, and Purposed.*

- ***Cost:*** The cost is one of the most significant challenges in implementing traceability, particularly as the system grows in size and sophistication, making requirement tracing more expensive and difficult [13].

- ***Obtaining Information***: Difficulty in finding the data that is required to assist and improve the tracing process [13].

- ***Organizational Problems:*** Some organizations do not provide their employees with training on the importance of traceability in processes, which can result in a lack of representation and accountability. [14].

- ***Purpose:*** This challenge implies that traceability should be established with a specific reason in mind, ensuring it aligns with the needs of various stakeholders both within and outside the company. [14].

C. Social Challenges

Social challenges include different stakeholder viewpoints, politics and lack of training, lack of communication between groups, trust issues, and constantly changing requirements.

- ***Different stakeholder viewpoints:*** Traceability may not be given the attention it deserves in some parts of an organization, which will lead to an inefficient allocation of time, personnel, and resources. To create an organizational policy for traceability that can be consistently applied to all projects within the company, it is currently best practice to consider the opinions of various stakeholders. Establishing an organizational policy on traceability is the best strategy to handle the various perspectives of stakeholders [14].

- • ***Valued*****:** The significance of strategy is recognized by everyone, and stakeholders actively participate in its creation within the company. However, if traceability is not regarded as important, it may be considered optional and given low priority, leading to an inadequately developed strategy [14].

- ***Politics and lack of training***: A factor contributing to poor support for traceability is politics and a lack of training. Some businesses fail to educate their personnel on the value of traceability [14].

- ***Trusted***: Establishing and maintaining trust and confidence in traceability is crucial among all stakeholders, especially in the face of inconsistencies, omissions, and changes [14].

- ***Requirements constantly change***: Constantly changing requirements result in the project becoming more complex and increasing the cost and effort involved [15].

- ***Lack of communication between groups:*** the failure of cooperation between the groups responsible for coordinating traceability illustrates the lack of communication between them. [16].

D. Technical Challenges

Technical challenges include *Configurable, Complexity, Portable, Poor tool support, Scalable*, and Ubiquitous.

- ***Configurable: "***Traceability is established as specified, moment-to-moment, and the rich semantics accommodate changing stakeholder needs." [34].

- ***Complexity***: Many factors, including the project's size, the sheer volume of objects that must be traced, and the challenging connections between these artifacts that can hinder traceability, can all be used to indicate complexity [34].

- ***Portable:*** Traceability information should be portable, meaning that it can be merged, exchanged, and reused across organizations, domains, projects, product lines, and tools supporting it [34].

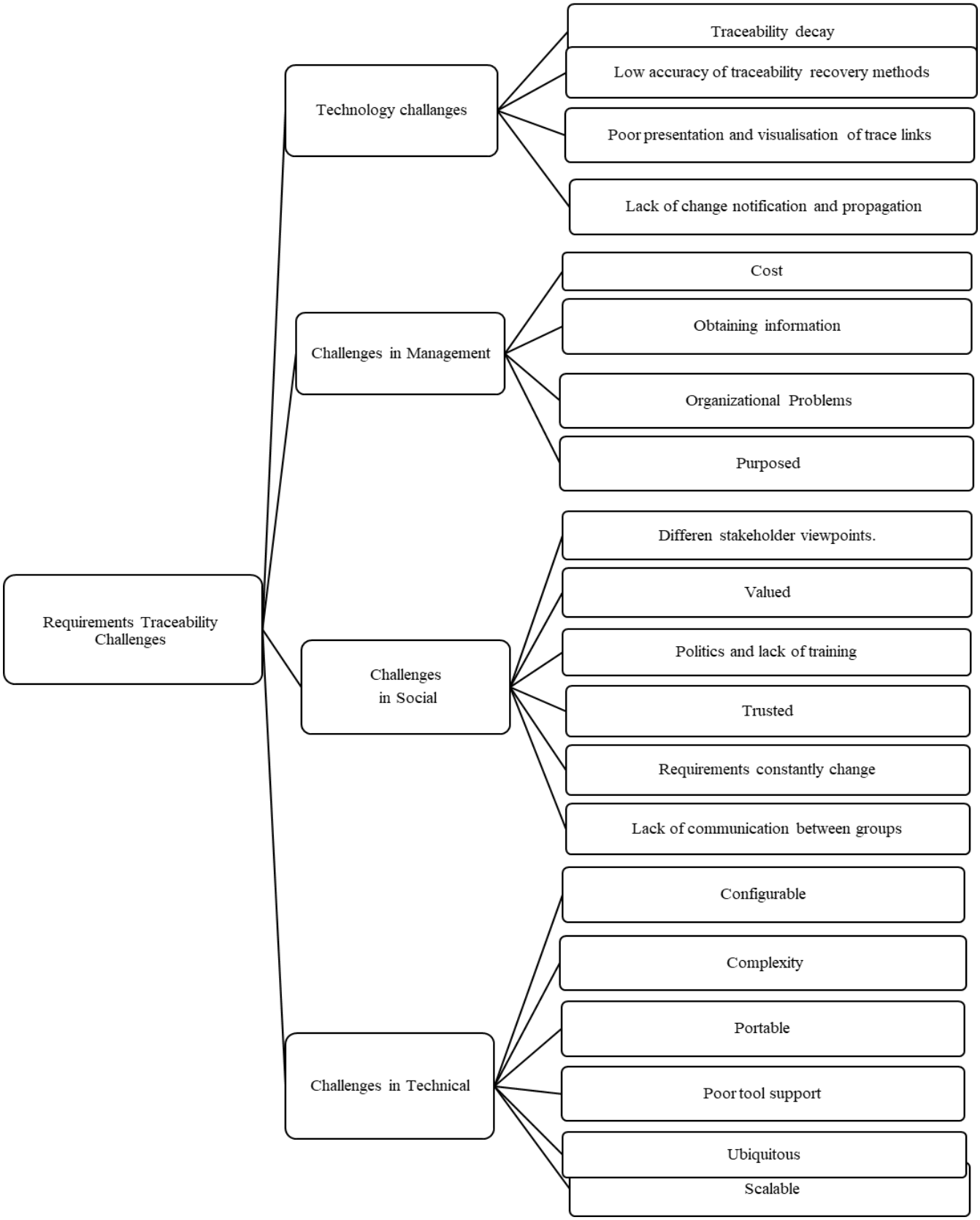

Fig 6: Requirements Traceability Challenges

- ***Poor tool support:*** Perhaps the largest obstacle to adopting traceability is inadequate tool support. There are numerous challenges with tools that might make establishing and maintaining traceability difficult. These include selecting from among available tools, the lack of standalone traceability tools, integration issues with other tools, and the difficulty of configuring a general-purpose tool or developing a custom tool [18].

- ***Scalable:*** Traceability should be scalable, meaning that it can support an increasing number of different types of artifacts with varying levels of granularity throughout the system's lifecycle and across the boundaries of the organization and business [34].

- ***Ubiquitous:*** The grand challenge in requirements traceability is achieving ubiquity, which means that traceability links should be automatically created as stakeholders and developers work, without requiring them to explicitly think about creating such links**.** [34].

2. State-of-the-art approaches utilized for requirements traceability (Question 2)

- This subsection discusses state-of-the-art approaches used for requirements traceability and their associated concerns from 2006 to 2019. These methods are applied to track both functional and non-functional requirements, automate the creation of traceability links, gather early feedback, improve the direction of requirement traceability [17], and help users manage their traceable items more effectively [18]. Table 5 identifies significant approaches and relevant concerns extracted from various studies, which can contribute to the development and enhancement of requirements traceability. The approaches include rule-based approaches [17] [19], value-based requirements tracing [18], information retrieval (IR) [19], event-based traceability [20] [21] [22], and goal-centric traceability (GCT) [39].

TABLE 5. REQUIREMENT TRACEABILITY APPROACHES

Approach Name	Relevant Concerns
Rule-based approach.	It focuses on gradual modifications to a developing network of traceability relationships. [25][35].
Value-based Requirements Tracing.	Performance evaluation should consider tracing precision and effort, which are determined by stakeholder value, requirements risk, and tracing costs [19].
Information retrieval.	Addressing the low precision issue, aim to enhance both precision and recall of traceability links [21][26]
Event-based Traceability	Understand the impacts of requirement changes in large systems [22].
Goal-centric traceability	Inability to trace Non-Functional Requirements (NFRs) such as security, performance, and usability [27] [39].

Table 6 presents the approaches to RT along with their relevant concerns, which have been derived from specific references.

3. The criteria used to evaluate traceability approaches (Question 3)

In summary, these criteria serve as indicators of how well these approaches support requirements traceability. They are aligned with the foundational elements used to assess the effectiveness of traceability recovery approaches in traditional software engineering practices [24].

Our Systematic Literature Review (SLR) has identified various traceability criteria used to evaluate traceability approaches, including database support [29], software traceability support, tool support, coverage construction [24], traceability scheme, scalability [49], visualization, change impact analysis, evaluation [27], trace type [24], trace direction [24], traceability analysis [24], underlying technique, and accessibility [24]. Table 6 provides a concise summary of these criteria along with their definitions. The ensuing criteria encapsulate the key factors influencing an effective traceability recovery approach for evaluating any conventional software engineering method. These criteria have been derived from an extensive analysis of literature and are inspired by insights from influential researchers in the field. The subsequent section provides a brief explanation of these criteria [24].

Table 6 provides a succinct overview of these criteria alongside their definitions. Figure 7 visually represents the criteria discussed in this Systematic Literature Review (SLR), which have been employed in assessing Real-Time (RT) approaches based on prior studies. These criteria are applicable for evaluating any traceability approach within traditional software engineering. They have been formulated through a comprehensive examination of literature (e.g., [17], [24], [29], [49]), as well as inspired by insights gleaned from eminent researchers in the field.

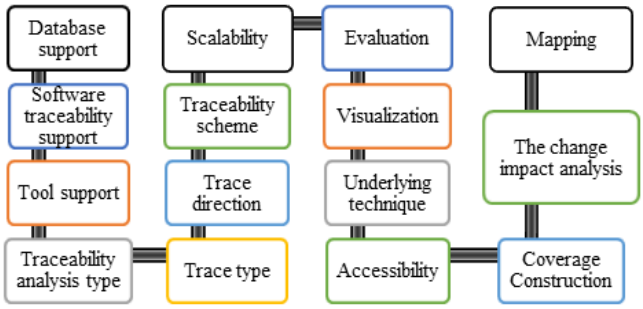

Fig 7: Criteria used to Evaluate Traceability Approaches

4. The major characteristics of the identified RT approaches (Question 4)

This section provides a summary of the results obtained from the comprehensive analysis of selected papers to address RQ4, outlined in Table 7 and Figure 8. Additionally, we have delineated the key characteristics of the requirements traceability (RT) approaches identified in response to the second research question.

TABLE 6. CRITERIA USED TO EVALUATE TRACEABILITY APPROACHES

Criteria Name	Criteria Definition
Database support	Utilized to record code elements and test coverage information for later retrieval [29].
Software traceability support	It is used in order to link software artifacts (Code, Requirement, and Test Case) in two ways: requirement traceability (Requirement to Test Case) and code traceability (Code to Test Case) [29].
Tool support	It is used for facilitating traceability links and automatically monitors the status of all change requests and notifications to ensure that the necessary follow-up actions are carried out as needed [24].
Traceability analysis type	Traceability analysis encompasses three types: manual, semi-automatic, and automatic [24].
Trace type	It is categorized into two main groups: Functional traces and Non-functional traces [24]
Trace Direction	A traceability link can be unidirectional (like 'depends-on') or bidirectional (like 'alternative-for') [24].
Traceability scheme	A traceability scheme identifies the artifacts involved in the trace link recovery process and specifies the granularity level for recording trace links for each artifact. [24].
Scalability	"It is important that traceability approaches be scalable for both capturing links and presenting the linked information to users. In manual or minimally-automated settings, scalability is more achievable due to the incremental capture and maintenance of trace links." This suggests that the approach can be effectively applied to large projects.[24].
Evaluation	The function of this criterion is to determine how the approach was evaluated in the original work, such as through controlled experimental studies, case studies, or survey studies [24]
Visualization	It facilitates users in evaluating the quality of trace links, underscoring the significance of traceability recovery methods that provide insightful graphical representations for visualizing traces and identifying outdated or questionable links [24].
Underlying technique	"It represents the core of an approach, through which traceability links between tests and code are recovered." [24].
Accessibility	"Mapping between artifacts in software development life cycle" [17].
Coverage Construction	The assessment of requirements traceability approaches also hinges on coverage construction, a crucial criterion. This assessment pivots on whether coverage is attained through the execution of test cases or otherwise [24].
The change impact analysis	Specifies whether the approach determines the impact of change on the artifacts throughout the software development lifecycle [49].
Mapping	Assess whether the approach can establish links between models at varying levels of abstraction [24].

TABLE 7. REQUIREMENTS TRACEABILITY APPROACH

RT Approach Name	Ref.
Rule Based Approach (RB)	[21][46][47]
Value-Based Requirements Tracing (VB)	[22][23]
Information Retrieval (IR)	[16][19][30][31][33][36][37] [38] [40][41][42][43]
Event-based Traceability (EB)	[23][29]
Goal-Centric Traceability (GC)	[39][44][48][50]

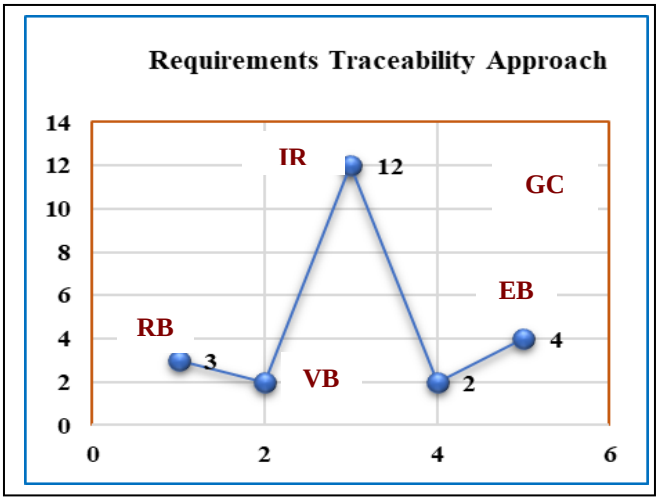

Fig 8. Requirements Traceability Approach

We note from Table 7 and Figure 6 that the Information Retrieval approach to requirements traceability has more references than alternative methodologies such as the Rule-Based Approach, Value-Based Requirements, Event-based Traceability, and Goal-Centric Traceability. Conversely, the Value-Based Requirements Tracing approach and Event-based Traceability exhibit fewer references compared to their counterparts. This observation suggests that from 2006 to 2019, there were relatively fewer studies referencing these five requirements traceability approaches. To stimulate further research, authors may consider extending the timeframe. The subsequent section delineates the principal characteristics of the identified requirements traceability approaches.

- **Rule-Based Approach.**

The rule-based approach demonstrates several notable characteristics, including its capability to export all supported artifacts into XML format and the utilization of rules for generating traceability relations based on the exported state of the models. Additionally, it sustains post-requirements traceability relations, facilitating the automatic generation of traceability links among various artifacts, such as requirements, use cases, and analysis object models [46]. In the Rule-based approach, two types of rules are at work. **The first** type involves requirement-to-object-model rules, which utilize information retrieval techniques to automatically establish traceability relations between requirements and analysis models. **The second type** of rule evaluates links between requirements and object models,

identifying intra-requirement dependencies and automatically establishing these relations [21].

The Requirement-to-Object Model rules, combined with an information retrieval technique, are instrumental in creating traces. This approach seeks to address the issue of insufficient automated support for maintaining traceability relations within UML-based development tools, often stemming from the activities conducted within these tools [46]. Its primary focus lies in automating the preservation of traceability relations between two artifacts. To accomplish this, it utilizes a prototype tool called 'traceMaintainer' to recognize development activities applied to models within UML-based software development. Furthermore, it acts as a solution to mitigate traceability decay [47].

- **Value-Based Requirements Tracing Approach (VBRT):**

This approach integrates both semi-automated and manual methods to establish traceability links and implement changes in software artifacts. The tracing scope encompasses source code, design elements, and requirement documents, facilitating the identification of link traces according to prioritized requirements [22]. It consists of five processes: (1) requirements definition, which involves identifying atomic requirements and giving each one a unique identifier; (2) requirements prioritization, which involves determining the value, risk, and effort of each requirement; (3) requirements packaging, which involves identifying clusters of requirements; (4) requirements linking, which involves creating traceability links between requirements and other artifacts; and (5) evaluation, which involves determining the effectiveness of the approach [22], as shown in Figure 9.

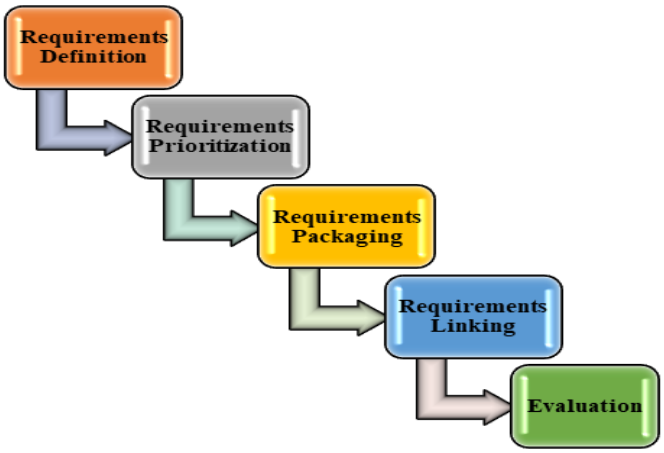

Fig 9. VBRT Processes

The VBRT approach provides both a technical and an economic model for requirements tracing, tailored to various criteria including the quantity, value, and risk of requirements, as well as factors such as the volume of artifacts, the extent of traceability, precision in tracing, artifact size, and associated costs and efforts in tracing identification and maintenance [23]. The primary objective of this approach is to discern the significance and value of individual requirements [23]. Figure 10 illustrates the criteria utilized within the VBRT approach.

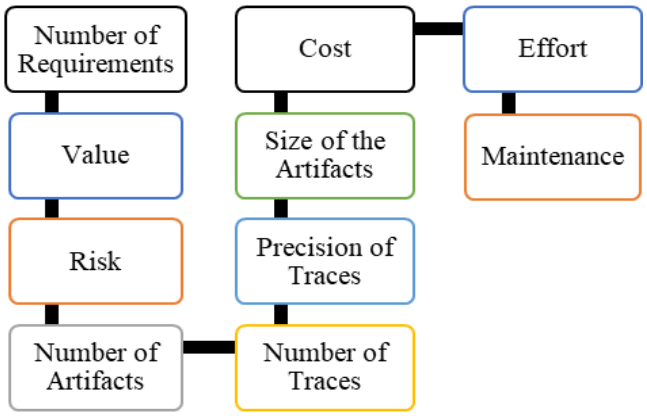

Fig 10. The Criteria of the VBRT Approach

- **Information Retrieval (IR)**

The Information Retrieval (IR) approach primarily focuses on automating the creation of traceability links by comparing two distinct artifact types, such as requirements and source code [33]. IR "focuses on finding documents whose content matches with a user query from a large document collection" [51]. Information retrieval (IR) techniques are frequently applied in the field of software engineering. They support various tasks, including document reuse and tracing between two or more documents [52].

IR methods, widely recognized and applied, offer an effective solution for recovering traceability links between various artifact pairs, including requirement documents and source code [16]. These methods are known for expediting the generation of traceability links, thereby reducing the time required for establishing traceability mappings [40]. The most popular methods in Information Retrieval (IR) adapted for requirements traceability recovery include Latent Semantic Indexing (LSI) [16], the Vector Space IR Model (VSM), and IR Probabilistic. LSI, for instance, enhances system understanding by reconstructing traceability links among various artifacts generated during development [41]. VSM, originally developed for the SMART information retrieval system [42], and Probabilistic methods compute ranking scores based on the probability that a document (the target artifact) is related to the query (the source artifact) [43]. These methods establish links between different artifacts primarily based on their syntactic information [36]. The management of links between two artifacts begins with their creation and is facilitated using suitable tools [37].

IR-based traceability recovery approaches generate ranked lists of traceability links between segments of requirements and source code. After that, these links are refined using various pruning strategies and subsequently validated by human experts [19]. Figure 11 illustrates the IR-based recovery process. IR systems strive to establish a connection between users' information needs and the information within a document collection. The fundamental information retrieval process consists of two phases:

- **Indexing**: In this phase, the provided information within the documents is stored and organized.
- **Similarity Computation**: Here, a similarity score is calculated between a user query and the documents stored in the index [30].

Classic IR methods, such as the Vector Space Model (VSM), offer relevance ranking for a given query but overlook document organization, supporting only flat queries. Furthermore, they operate on static documents, usually retrieving entire documents as units of relevance [30].

The primary challenge in Information Retrieval (IR) lies in identifying the pertinent documents within a document set according to user-defined information requirements. Many IR approaches involve converting each document in the set into a mathematical representation that encapsulates its informational essence. Then, a comparison is made with similar representations of user information needs (queries) [31].

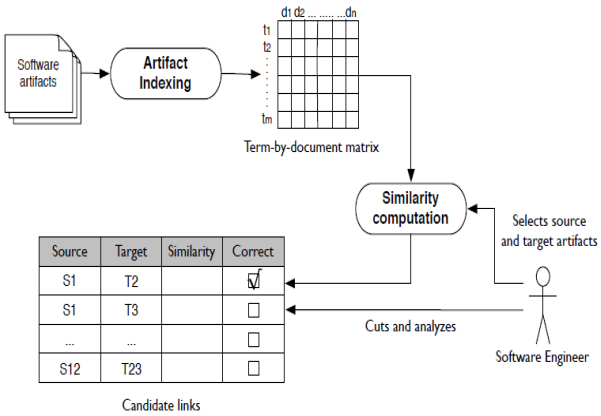

Fig 11. IR Recovery Processes[38]

- **Event-based Traceability (EBT)**

The EBT approach employs an event-based traceability model to effectively manage changes and deliver timely updates for affected artifacts along with their associated links [23]. In this methodology, traceability relationships are established as publisher-subscriber relationships, where reliant artifacts subscribe to the requirements they depend on. Whenever a requirement undergoes modifications, the dependent artifacts receive notifications, enabling them to take necessary actions [29]. The approach comprises three core components:**1. Requirement Manager**: This component is responsible for overseeing requirements and issuing change event messages to the event server.**2. Event Server:** The event server manages initial subscriptions made by dependent entities, processes event notifications from requirement managers, and forwards event messages to pertinent subscribers. **3. Subscribing Entity:** These are the entities that subscribe to and receive notifications concerning changes [29], as illustrated in Fig 12.

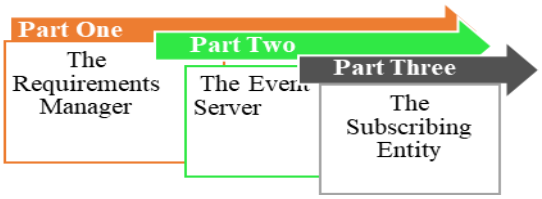

Fig 12. EBRT Approach Parts

- **Goal-Centric Traceability.**

Goal-centric traceability (GCT) emphasizes quality goals, typically specified as Non-Functional Requirements (NFRs) or constraints [39]. The Systems and Requirements Engineering Center (SAREC) at DePaul University has employed various tools to implement GCT. Poirot, a robust tracing tool, utilizes a probabilistic network to generate traceability links on demand, connecting design and code artifacts to nodes within a design hierarchy. Additionally, students in the MS Studio class developed a graphical tool for creating models of soft goal interdependency graphs (SIGs). Previously, the University of Illinois at Chicago developed an event-based traceability (EBT) utility that establishes links between QAMs and goals. Furthermore, an XML-based utility was designed for graph traversal and the reevaluation of QAMs [39].

Goal-Centric Traceability (GCT) serves the purpose of tracking Non-Functional Requirements (NFRs) through three primary activities: requirement development, impact detection, and evaluation and decision-making [44]. Across the prolonged lifespan of a software-intensive system, GCT furnishes traceability support for the management and upkeep of NFRs along with their correlated quality concerns [48]. This methodology heightens the automation for leveraging and comprehending traceability links. While it is tailored with maintainability in mind in certain scenarios, it necessitates specific modeling environments or development practices. Moreover, as it isn't directly integrated into specific tasks such as architectural analysis, it might not immediately yield advantages for trace creators. This raises pragmatic concerns regarding the cost-benefit ratio associated with investing in a traceability infrastructure, potentially impeding its widespread adoption in practical contexts [48].

The GCT framework comprises the following elements: (1) a goal model capturing stakeholders' quality concerns and tradeoffs; (2) a series of Quality Assessment Models (QAMs) designed to evaluate how effectively the architecture meets the defined quality objectives; (3) a

traceability system linking QAMs to goals; (4) GCT algorithms overseeing automated impact analysis and change propagation throughout the goal hierarchy; and (5) an impact report depicted in Fig 13. GCT facilitates the establishment of trace links around various architectural decisions and facilitates essential software engineering tasks such as analyzing stakeholder satisfaction, validating requirements, preserving architecture, conducting impact analysis, and constructing safety cases [50].

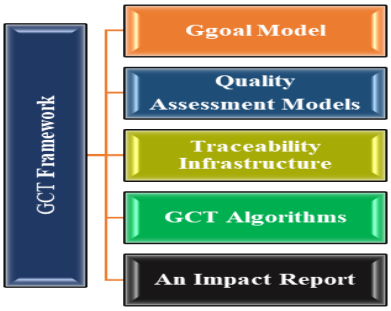

Fig 13. GCT Framework

Table VIII presents the various approaches for requirements traceability discussed above, showcasing how each approach handles the tracing of both Functional Requirements (FR) and Non-Functional Requirements (NFR).. "Functional requirements describe the functional behavior of the system whereas nonfunctional requirements" [53].

TABLE 8. FUNCTIONAL AND NON-FUNCTIONAL TRACING APPROACH

Name of Approach	Type of Requirements	
	FR	NFR
Rule Based Approach	✓	//////////
Value-Based Requirements Tracing Approach	✓	✓
Information Retrieval Approach	✓	✓
Event-based traceability Approach	✓	✓
Goal-Centric Approach	/////////	✓

Drawing from the responses to the four research questions, our findings reveal a notable scarcity of research focused on requirements traceability approaches. This underscores the imperative for additional research endeavors to delve into and enrich the landscape of requirements traceability methodologies.

VI. CONCLUSION AND FUTURE WORK

Requirements traceability (RT) stands as a significant quality factor in software development, enabling software engineers to monitor requirements from inception to fulfillment. This study presents a Systematic Literature Review (SLR) in the field of requirements traceability approaches. We meticulously followed the guided steps of SLR, commencing with planning the review phase (See Section 3.1), proceeding to conducting the review phase (See Section 3.2), and culminating in reporting the review phase (See Sections 4 and 5).

Our contribution encompasses the classification of requirements traceability challenges and a succinct overview of state-of-the-art approaches. We also provide criteria for evaluating traceability approaches and outline the key characteristics of identified requirements traceability approaches from 2006 to 2019. These criteria and characteristics serve as invaluable guidance for researchers and practitioners seeking specific approaches of interest.

Consequently, our findings augment the body of knowledge concerning the most significant challenges of requirements traceability encountered by traceability approaches. Grasping these challenges can aid software engineers and developers in enhancing their work and achieving superior traceability by selecting suitable approaches or refining existing ones. Furthermore, we furnish a brief overview of state-of-the-art approaches, juxtapose the most crucial criteria for evaluating traceability approaches to assist in selecting suitable ones and discuss the principal characteristics of identified requirements traceability approaches utilized to fortify traceability. These topics have not been previously addressed together in previous SLRs.

Our future research goal is to explore new criteria, focusing on innovative approaches and utilizing new tools. This will involve extending the research period to include data up to 2024 and incorporating additional digital libraries

VII. REFERENCES

[1] A.Kannenberg, & H. Saiedian. "Why software requirements traceability remains a challenge". *CrossTalk The Journal of Defense Software Engineering*, vol.22.no.5,pp. 14-19, 2009.

[2] J. Cleland-Huang. "Just enough requirements traceability". *In 30th Annual International Computer Software and Applications Conference (COMPSAC'06)*; Vol. 1, pp. 41-42. *Chicago. IEEE*.2006.

[3] R.Torkar,T. Gorschek,R.Feldt,M. Svahnberg, U.A. Raja, & K. Kamran. "Requirements traceability: a systematic review and industry case study". *International Journal of Software Engineering and Knowledge Engineering*, pp.385-433.2012.

[4] I.Santiago,A. Jiménez,J.M. Vara,V. De Castro, V. A. Bollati, &E. Marcos. "Model-Driven Engineering as a new landscape for traceability management: A systematic literature review". *Information and Software Technology*. Vol.54,no.12, pp. 1340-1356,2012.

[5] M. A Javed, & U. Zdun. "A systematic literature review of traceability approaches between software architecture and source code" . *In Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering* .p.16. Vienna, Austria: ACM.2014.

[6] H.Tufail, M.F.Masood, B. Zeb,F.Azam, & M.W.Anwar. " A systematic review of requirement traceability techniques and tools". *2nd International Conference on System Reliability and Safety* (ICSRS) *IEEE*; pp. 450-454,vol.12 . 2017

[7] B.A.Kitchenham & S.Charters, "Guidelines for Performing Systematic Literature Reviews in Software Engineering". *Technical Report EBSE*, vol.01, pp.1-65.2007.

[8] A.M. Salem. "Model for Enhancing Requirements Traceability and Analysis. International Journal of Advanced Computer Science and Applications", vol. 1, no.5, 2010.

[9] J.Cleland-Huang, A.Czauderna, A.Dekhtyar,, O.Gotel,J.H.Hayes,E.Keenan,& A.Zisman, "Grand challenges, benchmarks, and TraceLab: developing infrastructure for the software traceability research community". *ACM* .pp.17-23,.2011.

[10] G.Regan,F. McCaffery,K. McDaid,& D. Flood. "The barriers to traceability and their potential solutions: Towards a reference framework". In Software Engineering and Advanced Applications (SEAA)*IEEE*. pp. 319-322. 2012.

[11] J. Guo, M. Gibiec, and J. Cleland-Huang, "Tackling the term-mismatch problem in automated trace retrieval. Empirical Software Engineering", 2016; Vol.22, No.3, PP.1103–1142.

[12] C.L.Williams. “A MANY-TO-MANY (M:N) RELATIONAL MODEL TO IMPROVE REQUIREMENTS TRACEABILITY.2011.

[13] S.Nair,J.L. De La Vara, &S.A. Sen. “review of traceability research at the requirements engineering conference re@ 21”. *In 21st IEEE International Requirements Engineering Conference* (RE) *IEEE*, Brazil. pp. 222-229.2013.

[14] S. Maro. “Addressing Traceability Challenges in the Development of Embedded Systems”.2107.

[15] M.Shahid,&S.A. Ibrahim. “New Model For Requirements to Code Traceability to Support Code Coverage Analysis”. *Asian Academic Research Journal of Multidisciplinary (AARJMD),*vol.1.no.14, PP.159-172.2013.

[16] D.V. Rodriguez, & D.L. Carver. “Comparison of information retrieval techniques for traceability link recovery”. *In 2019 IEEE 2nd International Conference on Information and Computer Technologies (ICICT)*IEEE. pp. 186-193.2019.

[17] A.Azmi, & S. Ibrahim. “Formulating a Software Traceability Model for IntegratedTest Documentation: a Case Study”. *International Journal of Information and Electronics Engineering*.vol. 1,no.2, 178.2011.

[18] N.Niu, W. Wang, &A. Gupta. “Gray links in the use of requirements traceability”.*In Proceedings of the 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering* , ACM.pp. 384-395.2016.

[19] N.Ali,Y.G. Guéhéneuc,& G. Antoniol. “Trust-based requirements traceability”. *In 2011 IEEE 19th International Conference on Program Comprehension. IEEE*.pp.111-120. 2011.

[20] R.P.Gohil,S.Bhattacharya,&R.Chauhan. “ Requirements Change Impact Analysis Using Event Based Traceability”. *Lecture Notes on Software Engineering*, vol. 4, no.3.pp. 162.2016.

[21] Rochimah, S., WAN KADIR P.Mäder, and G.Orlena. “Towards automated traceability maintenance”. *Journal of Systems and Software*, vol. 85, pp.2205-2227.2012.

[22] W. M., & A.H.Abdullah. “Utilizing multifaceted requirement traceability approach: a case study. International Journal of Software Engineering and Knowledge Engineering”, vol. 21.no.04,pp. 571-603.2011.

[23] T.Shereni. “Effective and efficient requirement traceability in the software development and Information Technology industry”. *(Doctoral dissertation, University of Cape Town)*2015.

[24] R. M. Parizi, , S. P. Lee, , &M. Dabbagh . “Achievements and challenges in state-of-the-art software traceability between test and code artifacts” . *Transactions on Reliability, IEEE*; vol.63.no.4, pp.913-926..2014.

[25] G.C.Filho,M. Lencastre,A. Rodrigues, & C.Schuenemann .RETRATOS: “Requirement Traceability Tool Support”. *semanticscholar.org*, VOL.6. 2013

[26] S. Murugappan, & D. Prabha. “Requirement Traceability for Software Development Lifecycle”. *International Journal of Scientific & Engineering Research*,vol. vol.8.no.5,pp. 1-11.2017.

[27] W.T.Lee,W.Y. Deng,J. Lee, &S.J. Lee. “Change impact analysis with a goal-driven traceability-based approach”. *International Journal of Intelligent Systems*,2010; 25(8), 878-908.

[28] P. Mäder, O. Gotel, & I. Philippow. “Rule-based maintenance of post-requirements traceability relations”. *In 16th IEEE International Requirements Engineering Conference, IEEE*. Spain. pp. 23-32.2008.

[29] M. Shahid, S.Ibrahim & H. Selamat. “Test coverage measurement and analysis on the basis of software traceability approaches”. *International Journal of Information and Electronics Engineering*, vol.1no.2, 115-119.2011.

[30] N. Sannier, &B. Baudry. “Toward multilevel textual requirements traceability using model-driven engineering and information retrieval”. *In 2012 Second IEEE International Workshop on Model-Driven Requirements Engineering (MoDRE)*. pp. 29-38. 2012

[31] Al-Saati, N., & Abdul-Jaleel, R. Requirement Tracing using Term Extraction. arXiv preprint arXiv:1506.08789.2015.

[32] P. Rempel, Mçder, P., & Kuschke, T. An empirical study on project-specific traceability strategies. In 2013 21st IEEE International Requirements Engineering Conference (RE) 2013; (pp. 195-204). IEEE.

[33] M.Hassnain. “A Comparative Study on Traceability Approaches in Software Development Life Cycle”. *ITEE Journal Information Technology & Electrical Engineering*, VOL.4.NO.2.2015.

[34] O.Gotel,J.Cleland-Huang, J., Hayes, J. H., Zisman, A., Egyed, A., Grünbacher, P., & Maletic, J. The grand challenge of traceability (v1. 0). Software and Systems Traceability, 343-409.2012.

[35] P. Mäder, O. Gotel, O., &I.I. Philippow, I. (SeHinptember). “Rule-based maintenance of post-requirements traceability relations”. *In 2008 16th IEEE International Requirements Engineering Conference.*; pp. 23-32. IEEE.

[36] A. Mahmoud, & N. Niu, N. Using Semantics-Enabled Information Retrieval in Requirements Tracing: An Ongoing Experimental Investigation”. IEEE 34th Annual Computer Software and Applications Conference. VOL.29.2010

[37] E. Bouil lon, P. Mäder, & I. Philippow, “A survey on usage scenarios for requirements traceability in practice. In Requirements Engineering: Foundation for Software Quality”: 19th International Working Conference, REFSQ, Essen, Germany, vol 8-11, 2013. Proceedings 19 (pp. 158-173). Springer Berlin Heidelberg.

[38] G. Capobianco, A.D. Lucia, R. Oliveto, A. Panichella, &S. S. Panichella. “Improving IR-based traceability recovery via noun-based indexing of software artifacts”. *Journal of Software: Evolution and Process*, VOL.25. NO.7,PP. 743-762.2013.

[39] J. Cleland-Huang, W. Marrero, & B. Berenbach. (2008). Goal-centric traceability: Using virtual plumblines to maintain critical systemic qualities. IEEE Transactions on Software Engineering, vol.34, no.5, pp 685-699.2008

[40] S. Sundaram, J.H. Hayes, A, Dekhtyar, & E.A. Holbrook, E. A. Assessing traceability of software engineering artifacts. Requirements Engineering,2010;15(3), 313–335. doi:10.1007.

[41] M. Lormans, & A. Van Deursen, “A. Can LSI help reconstructing requirements traceability in design and test? In Conference on Software Maintenance and Reengineering (CSMR'06) IEEE. VOL.2. pp. 10. .2006.

[42] P. D. Turney, & P. Pantel, “From frequency to meaning: Vector space models of semantics”. *Journal of artificial intelligence research, vol.*37, pp.141-188.2010.

[43] A. De Lucia, A. Marcus, Oliveto, R., & R.D. Poshyvanyk, . Information retrieval methods for automated traceability recovery. Software and systems traceability, PP. 71-98.2012.

[44] M. Kassab, O. Ormandjieva, & M. Daneva, “A traceability metamodel for change management of non-functional requirements”. *In 2008 Sixth International Conference on Software Engineering Research,* Management and *Applications*. IEEE. pp. 245-254.2008.

[45] V.Sinha, V., Sengupta, B., & Chandra, S. Enabling collaboration in distributed requirements management. IEEE software, 2006; 23(5), 52-61.

[46] P. Mader, O. Gotel, O., & I. Philippow. “Enabling automated traceability maintenance by recognizing development activities applied to models”. *In 2008 23rd IEEE/ACM International Conference on Automated Software Engineering*. IEEE. pp. 49-58.2008

[47] M. Mirakhorli &J. Cleland-Huang. “Tracing non-functional requirements.” *In Software and systems traceability*. pp. 299-320. Springer, In Software and systems traceability . (pp. 299-320). 2012.

[48] D. Hanspeter, A. Janes, A. Sillitti, & G. Succi, “ Improving the identification of traceability links between source code and requirements”. In DMS. pp. 95-100.2012.

[49] I. Galvao, I., &A. Goknil. “Survey of traceability approaches in model-driven engineering.*In 11th IEEE International Enterprise Distributed Object Computing Conference*.pp. 313-313).Netherlands, IEEE.2007..

[50] J. Cleland-Huang, M. Mirakhorli, Czauderna, A., & Wieloch, M. “Decision-Centric Traceability of architectural concerns”. *2013 7th International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE*). 2013; doi:10.1109/tefse.2013.6620147

[51] A.R.Rivas,E.L. Iglesias, & L.Borrajo. “Study of query expansion techniques and their application in the biomedical information retrieval”. The Scientific World Journal, 2014.

[52] W.Wang, He, Y., Li, T., Zhu, J., & Liu, J. “An integrated model for information retrieval based change impact analysis”. *Scientific Programming*, PP.1-13.2018.

[53] M.Dabbagh, & S. Lee, S. P. (2014). “An approach for integrating the prioritization of functional and nonfunctional requirements”. *The Scientific World Journal*, 2014.