2017 IEEE 25th International Requirements Engineering Conference

Datasets from Fifteen Years of Automated Requirements Traceability Research

Current State, Characteristics, and Quality

Waleed Zogaan, Palak Sharma, Mehdi Mirahkorli
Software Engineering Department
Rochester Institute of Technology, USA
{waz7355,ps2671,mxmvse}@rit.edu

Venera Arnaoudova
School of Electrical Engineering and Computer Science
Washington State University, USA
venera.arnaoudova@wsu.edu

***Abstract*—Software datasets play a crucial role in advancing automated software traceability research. They can be used by researchers in different ways to develop or validate new automated approaches. The diversity and quality of the datasets within a research community have a significant impact on the accuracy, generalizability, and reproducibility of the results and consequently on the usefulness and practicality of the techniques under study. Collecting and assessing the quality of such datasets are not trivial tasks and have been reported as an obstacle by many researchers in the domain of software engineering. This paper presents a first-of-its-kind study to review and assess the datasets that have been used in software traceability research over the last fifteen years. It presents and articulates the current status of these datasets, their characteristics, and their threats to validity. Furthermore, this paper introduces a Traceability-Dataset Quality Assessment (T-DQA) framework to categorize software traceability datasets and assist researchers to select appropriate datasets for their research based on different characteristics of the datasets and the context in which those datasets will be used.**

***Index Terms*—Software Traceability, Datasets Quality, Traceability Automation, Systematic Literature Review**

I. Introduction

Advances in the area of automated requirements traceability research rely on the availability of different types of *datasets*. *Training sets* are needed to train trace-algorithms based on Machine Learning (ML) techniques. For instance, researchers have used a labeled dataset of functional requirements and non-functional requirements to train classification techniques to create traceability links between quality attributes and requirements document, design models and source code [1]–[5]. *Validation sets* are needed to tune parameters of such trace-algorithms [4], [6], [7]. *Testing sets* are used to test the performance of trace-algorithms on unseen data. For instance, researchers have used datasets to evaluate the accuracy of trace-algorithms based on Information Retrieval (IR) that establish links between requirements and source code [5], [8]–[11].

Obtaining such software development datasets has been one of the most frequently reported barriers for researchers in the software engineering domain in general [12], [13]. This problem is even more acute in the area of requirements traceability which is crucial in safety critical and highly regulated application domains [14]. Many of the publicly available open source systems are not representative of those domains and consequently may not be suitable to use for training, validation, or testing sets. Thus, a key challenge is to determine the quality of datasets used to conduct a study, to develop an automated technique, or to validate the results of research work. Furthermore, it is important to explicitly state the threats to validity associated with the datasets so that research results are articulated with perspective to the underlying datasets.

Despite the crucial role of trace datasets, few efforts have been taken to understand the characteristics and limitations of the datasets in the area of requirements traceability [15] as well as the threats to validity associated with the results obtained with these datasets. Similarly, few efforts have been taken to standardize how the datasets quality tracking and assurance should be implemented [16]. To the best of our knowledge, no previous work defines quality dimensions and metrics for traceability datasets.

In this paper, we perform a systematic literature review (SLR) to assess the current state of software traceability datasets that have been used by researchers in the community over the past fifteen years. Specifically, we investigate 1) the *characteristics* of those datasets, 2) ways to evaluate their *quality*, 3) the *threats to validity* associated with those datasets, 4) factors that are associated with their *reusability*, and 5) the *diversity of datasets* used in the community. Our study provides new insights on the characteristics of datasets used in our community, presents a new quality assessment framework to reason about traceability datasets, and reveals tacit information about a large number of datasets used in the community which can highlight the path for addressing the threats to validity of the research conducted in this area. We provide details regarding the studied papers and datasets

2332-6441/17 $31.00 © 2017 IEEE
DOI 10.1109/RE.2017.80

in an online appendix[1].

The reminder of this paper is organized as follows: Section II discusses related work. Section III details the research questions answered in this work and the methodology we followed to perform the SLR. Section IV describes our findings. Section V lists possible threats to validity to our work. Section VI concludes this work and outlines future research directions.

II. Related Work

In this section, we discuss two main groups of related work: SLRs in the domain of software traceability (Section II-A) and quality assessment of datasets (Section II-B).

A. SLRs in traceability

Several SLRs exist in field of software traceability [17], [18], [19], [14].

Borg et al. [17] conducted an SLR on Information Retrieval-based trace recovery that included 79 publications. However, this study mainly focused on the classification of publications based on the IR techniques used by the authors. In contrast, our study focuses on characterizing the datasets used in the domain of automated software traceability research. The authors discussed that most requirements document used by researchers had less than 500 requirements, and results were reported only using precision and recall. However, the SLR did not focus on studying the datasets used within the community, therefore, lacks insights in this regard.

Nair et al. [18] looked at 70 papers related to software traceability from the International Conference on Requirements Engineering (RE) and inspected various aspects of traceability. The study covered all studies published between 1993 and 2013 but the scope of the study is very limited, covering only papers published at RE. Regarding the datasets usage, the authors mentioned that out of 70 papers, 27 (38.7%) do not specify any details about the datasets. They reported a rising trend in the traceability field with increasing emphasis on quality of experimentation and academic-industrial partnership.

Santiago et al. [19] conducted an SLR on the impact of Model-Driven Engineering in traceability. Based on 157 studies that were published before 2011, they reported that storage, data related operations, and visualization are more widely studied aspects of traceability compared to exchange and analysis.

Cleland-Huang et al. [14] analyzed the earlier and current trends in the field of software traceability. They pointed out some intriguing future research questions concerning the cost-effectiveness, trust, scalability, portability, ubiquity, and visualization aspects of traceability techniques. They reported that there is a lack of datasets that contain multiple artifact types (e.g., requirements, design, code, test cases, etc.), which in turn leads to limited studies in the direction of automation of traceability link evolution.

In this work, we conduct an SLR that focuses on the characteristics of traceability datasets as well as their quality. None of the previous studies provide a quality assessment of datasets. Moreover, we investigate the reusability of traceability datasets, which is not investigated by the above SLRs.

B. Assessing the quality of datasets

Liebchen and Shepperd [12] conducted a systematic review of 23 papers to study accuracy-based data quality. They concluded by stating that data quality should be taken into account while selecting a dataset. Another area of focus should be in identifying and correcting noisy datasets along with taking into account the impact of these noisy datasets on the results of the studies that use them.

Bosu and MacDonell [20] analyzed papers concerning data quality and identified issues within them. They emphasized the importance of datasets quality in software engineering case studies. In accordance with previous studies [14], [17], [18], the authors highlighted the need for industrial collaboration to understand and improve data quality.

Liebchen and Shepperd [12] looked into noisiness of datasets while Bosu and MacDonell [20] developed a taxonomy of data quality challenges in Empirical Software Engineering, emphasizing the importance of dataset quality. In this work, we are proposing a data quality framework tailored to the field of software traceability.

Wang and Strong [21] conducted a market research survey to derive a data quality framework adhering to consumers needs. They broadly categorized their subdimensions and data quality metrics in four labels: intrinsic, contextual, representational, and accessibility.

Zaveri et al. [22] surveyed 30 existing data quality assessment approaches and derived 18 dimensions and 69 metrics for assessing the quality of linked data. They also draw a comparative analysis of 12 tools using eight different attributes.

While the survey by Wang and Strong [21] is generic, the survey by Zaveri et al. [22] is focused on linked data. We complement those works by proposing a novel Traceability-Data Quality Assessment (T-DQA) framework, that is tailored to the field of traceability and that is based on several statistical metrics.

III. Research method

To identify and collect the datasets used in the software traceability community, we conducted a systematic literature review (SLR) of all published full papers with *empirical* and *automated* software traceability *theme*. We followed the guidelines that were established by

[1]Online appendix: https://goo.gl/aUUFpb

Kitcheman et al. [23] for SLR in Software Engineering. In this work, by traceability datasets we mean any form of data used by traceability researchers such as training set, testing sets, validation set, answer set, and case studies.

In the followings we present our research questions (Section III-A), the search strategy that we have used to identify relevant publications (Section III-B), the inclusion and exclusion criteria that we used (Section III-C), the overview of the paper selection process (Section III-D), and finally, the details of the data extraction approach from each paper (Section III-E) that allowed us to answer the research questions.

A. Research questions

- *RQ1: What are the characteristics of traceability datasets?*
  To answer this research question we define the following sub-research questions:
  - *RQ1.1: What are the source and target artifacts in traceability datasets?* We will collect the source and target types of the datasets and we will summarize the results by considering all traceability links types as bi-directional.
  - *RQ1.2: Which application domains are represented by traceability datasets?* We will identify the domain of each dataset and we will group the datasets based on their domains.
  - *RQ1.3: What is the size of traceability datasets?* To provide a standardized way of reporting size, we use a metric called *trace space* that provides a proxy for the complexity of a dataset. Trace space, D, is defined as the product of the size of the source and the size of the target artifacts:

    $$D_{TraceSpace} = |D_{Source}| \times |D_{Target}| \quad (1)$$

    Note that trace space defines the maximum number of trace links between two artifacts.
  - *RQ1.4: What proportion of the traceability datasets is from industry, open-source projects, and student generated data?* We will use frequencies to answer this research question.
  - *RQ1.5: Are traceability datasets available for reuse?* We will investigate whether the datasets are available online.
- *RQ2: How to assess the quality of traceability datasets?* We will investigate existing frameworks for evaluating datasets and we will adapt those to traceability datasets.
- *RQ3: Is there a relation between the characteristics and the quality of traceability datasets on the one hand and their reusability on the other hand?* To answer this research question we use Random Forest, a machine learning algorithm used for creating classification and regression trees. Random Forest can be also used for ranking the importance of different features [24]. The process of building a tree is iterative where the goal at each node is to split the data by using one variable only that best differentiates the data with respect to the dependent variable, i.e., create two nodes that are more homogeneous or more pure than the original node. Node purity is calculated using the residual sum of squares. We use the total decrease of node impurity, $IncNodePurity$, as an indication of the variable importance. A higher value of $IncNodePurity$ indicates more important input variable.
- *RQ4: What are the threats to validity associated with traceability datasets?* We will categorize and summarize the threats to validity related to the usage of datasets, acknowledged or mitigated by the studied papers.
- *RQ5: Do we, as a community, strive for a diversity of traceability datasets?* This would provide an insight on whether the same datasets are used for all the research problems in hand or whether we seek to adopt new datasets for different research problems. For authors that have published more than one papers, we calculate a ratio that represents the diversity of the datasets that they use. The diversity ratio for author i, $DiversityRatio_i$, is defined as the number of unique datasets, $UniqueDataset_i$, divided by the total number of datasets, $TotalDatasets_i$, used across the publications of author i.

  $$DiversityRatio_i = \frac{UniqueDataset_i}{TotalDatasets_i} \quad (2)$$

B. Search strategy

A search strategy is fundamental for any SLR to ensure that all relevant studies are considered for accurate conclusions [23], [25]. Our search strategy consists of the following main elements: *search methods*, *search terms*, and *data sources*. We performed a preliminary search to retrieve existing literature reviews in the domain of software traceability. We found a few SLRs that we discuss in Section II-A but none of them address the research questions that we defined.

We conducted our search using both *manual* and *automatic* methods to ensure that we cover as many relevant venues and electronic data sources as possible [25]. In the *manual search*, we went through all papers published in the venues listed in Table I. We built an initial list of relevant venues that we augmented by contacting traceability experts for suggestions on other related sources (conferences, journals, research groups active in this domain or individual papers). Table I contains venues that are considered high quality venues for software requirements (e.g., RE, TEFSE, and REJ), while other venues have a broader and generic theme (e.g., ICSE, TSE, and TOSEM). In the *automatic search*, we defined

a set of search terms. We started with key terms used in traceability papers such as *requirements* and *traceability*. To be as generic as possible we expanded the search terms to include *software traceability* which resulted in our final query as follows: *(software OR requirement) AND traceability*. We used the search terms during the automatic search to search in the electronic data sources listed in Table II by matching the terms with the title, keywords, and abstract of each paper.

TABLE I: Venues used in the manual search phase.

Conferences	
ICSE	International Conference on Software Engineering
TEFSE	International Workshop on Traceability in Emerging Forms of Software Engineering
ASE	International Conference on Automated Software Engineering
ESEC	European Software Engineering Conference
FSE	International Symposium on Foundations of Software Engineering
SST	International Symposium on Software and Systems Traceability
RE	International Requirements Engineering Conference
REFSQ	International Working Conference on Requirements Engineering: Foundation for Software Quality
COMPSAC	IEEE Computer Society International Conference on Computers, Software and Applications
ICSM	International Conference on Software Maintenance
MSR	International Conference on Mining Software Repositories
WICSA	Working IEEE/IFIP Conference on Software Architecture
ICPC	International Conference on Program Comprehension
ECSA	European Conference on Software Architectures
Journals	
EMSE	Empirical Software Engineering Journal
TSE	IEEE Trans. on Software Engineering Journal
ISSE	Innovations in Systems and Software Engineering Journal
SEP	Journal of Software: Evolution and Process
TOSEM	ACM Trans. on Software Engineering and Methodologies
REJ	Requirements Engineering Journal

TABLE II: Databases used in the automatic search phase.

ID	Database	Web address
1	ACM DL	http://portal.acm.org
2	IEEE Explorer	http://www.ieee.org/web/publications/ xplore
3	SpringerLink	http://www.springerlink.com
4	ScienceDirect	http://www.elsevier.com

TABLE III: Inclusion and exclusion criteria.

Inclusion criteria	
I1	A Full paper.
I2	Focus on software (requirements) traceability.
I3	Proposed/used/evaluated an automated traceability technique.
I4	Used data-sets in their study.
Exclusion criteria	
E1	Position papers, short papers, tool demo papers, keynotes, reviews, tutorial summaries, and panel discussions.
E2	A study that is not written in English.
E3	Duplicated studies.
E4	No datasets or case studies.

C. Inclusion and exclusion criteria

The inclusion and exclusion criteria are specified in Table III and were applied at different stages to all of the retrieved studies (See Figure 1). To limit the scope of our SLR, we included all studies that were published between 2000 and 2016. We have selected all studies that have used datasets, case studies or empirical data to develop, validate, train, or test traceability techniques. Papers that only presented approaches or an idea without empirical data-based validation were excluded. We considered only studies that focus on automated software traceability, therefore, papers related to models and processes of software traceability were excluded. In addition, we excluded short papers, workshops, and tool demonstration papers. Lastly, all duplicated studies found from different sources were identified and removed.

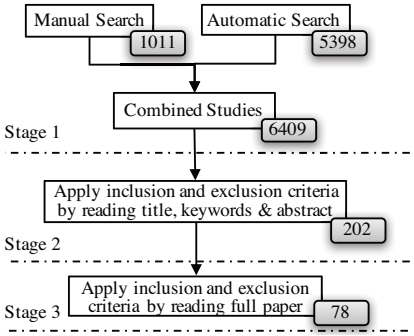

Fig. 1: Search process stages.

D. Study selection process

Figure 1 shows the number of studies selected at each stage of the SLR. The initial search process resulted in 1011 and 5398 papers that were collected during the manual and automatic search, respectively. Because of a large number of retrieved papers (6409), we selected the first set of papers that could be relevant to our study by reading their title, keywords, and abstract [26]. From 6409 papers, we selected 202 papers as the primary studies. All these 202 papers were reviewed by two of the authors of this paper. In this review process, the inclusion/exclusion criteria were applied and the rationale for these decisions was documented. The rationale for including/excluding the studies were reassessed and discussed in separate group discussions. If a paper satisfied all inclusion criteria, it was considered as a *primary study* and was included in SLR. Our final list included 78 papers.

TABLE IV: Extracted dataset items.

Data item	Related RQs
The **Name** & **Traceability Artifacts**	RQ1, RQ1.1
The **Application Domain**	RQ1, RQ1.2
The **Size (trace space)**	RQ1.3, RQ2
The **Type** (private, student, or open source)	RQ1.4, RQ2
The **Availability** and link to the source	RQ1.5, RQ2
The **Licensing, Storage, Developer, Programming Language**	RQ2, RQ3
The **Threats to validity** related to the datasets that are acknowledged or mitigated in the paper.	RQ4

E. Data extraction

In the data extraction phase, we collected all information from the selected studies that was necessary to answer our research questions. First, we extracted basic information about the papers such as the list

of the author(s), year of publication, title, venue, and publication type (full paper, short paper, etc.). Then, through a set of group studies, we identified, extracted, and organized the information about the datasets used in each research paper. Table IV provides an overview of the dataset items that were extracted from each paper and the research questions that require those items.

IV. Results

In this section, we summarize the results of our study and answer our research questions. From all papers studied in this SLR, we identified 73 unique datasets. For each of these datasets, we collected the data items listed in Table IV.

A. RQ1: What are the characteristics of traceability datasets?

This question is investigated through five sub-questions described below. Each sub-question examines different characteristics of traceability datasets.

RQ1.1: What are the source and target artifacts in traceability datasets?

Figure 2 shows the types of artifacts covered by the *73* datasets. The inner layer represents the *source* artifact type and the outside layer represents the *target* artifact type. Artifacts from the inner and outside layers are colored identically when there is an association between them in the datasets. More details about these data points and their frequency can be found in our online appendix.

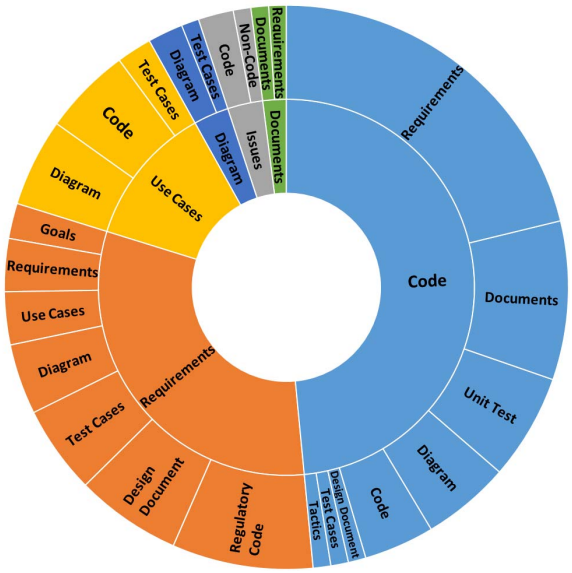

Fig. 2: Common *Source* and *Target* Artifacts.

All artifacts that specify textual requirement documents related to a dataset such as high level, low level, functional, and non-functional requirements are grouped under the "Requirements" category. In a similar fashion, the "Code" category consists of Java Classes, Code, Methods, and Classes. The "Test Cases" category groups all non-code test documents whereas the "Unit test" category is composed of the actual code implementations. The category "Document" is composed of artifacts such as manuals and other pages that datasets are being traced to.

As per our analysis, the most frequently, datasets are used to study traces between Code to: Code (4), Unit Test (6) and other Non-Code artifacts such as Requirements (21), Documents (9), Diagram (5), Design Document (1), Test Cases (1) and Tactics (1). Another category of commonly considered artifacts was found to be between "Requirements" and other Non-Code artifacts such as Design Document (6), Goals (2), Regulatory Code (8), Test Cases (5), Diagram (4), Requirements (3), and Use Cases (3). Less studied artifacts were Use Cases, Issue Reports, Diagrams, and Documents.

RQ1.2: Which application domains are represented by traceability datasets?

The frequencies and domains of the datasets are shown as a heat-map in Figure 3. Each colored block refers to an application domain. The sub-areas within a block represent a particular dataset where the area represents the frequency of that dataset usage.

Healthcare is by far the most frequent domain for traceability datasets [2], [6], [7], [9], [16], [27]–[48]. This is not surprising as traceability is crucial for safety critical and highly regulated domains [14]. Similarly, datasets from the *Aerospace* domain are frequently used by researchers [5], [7], [11], [15], [16], [30], [32], [33], [39], [41], [44], [49]–[56]. High proportion of datasets are also from the domains of *software engineering tools* [52], [57]–[67], *development libraries* [9], [15], [41], [60]–[62], [68]–[73], and *entertainment* [1], [3], [8], [15], [16], [28], [29], [32]–[34], [38], [42]–[44], [56], [74]–[79]. The majority of these systems are open source and available online which might explain their frequent usage by researchers. In addition, the researchers already serve as subject matter experts for some of these domains (e.g., software engineering tools or development libraries).

Industries such as *Power & Automation* have been used but less frequently [52], [67], [80]–[83]. All except one of the datasets from this domain are closed source.

RQ1.3: What is the size of traceability datasets?

Our analysis shows that there is an enormous gap in size among datasets that have been used by researchers (Table V). The minimum trace space size is from the industrial datasets and it is 42 while the maximum one is over 29 million and it is a software system in the *power and automation* domain, containing 4845 issue reports and 6104 non-code artifacts [80]. The median of

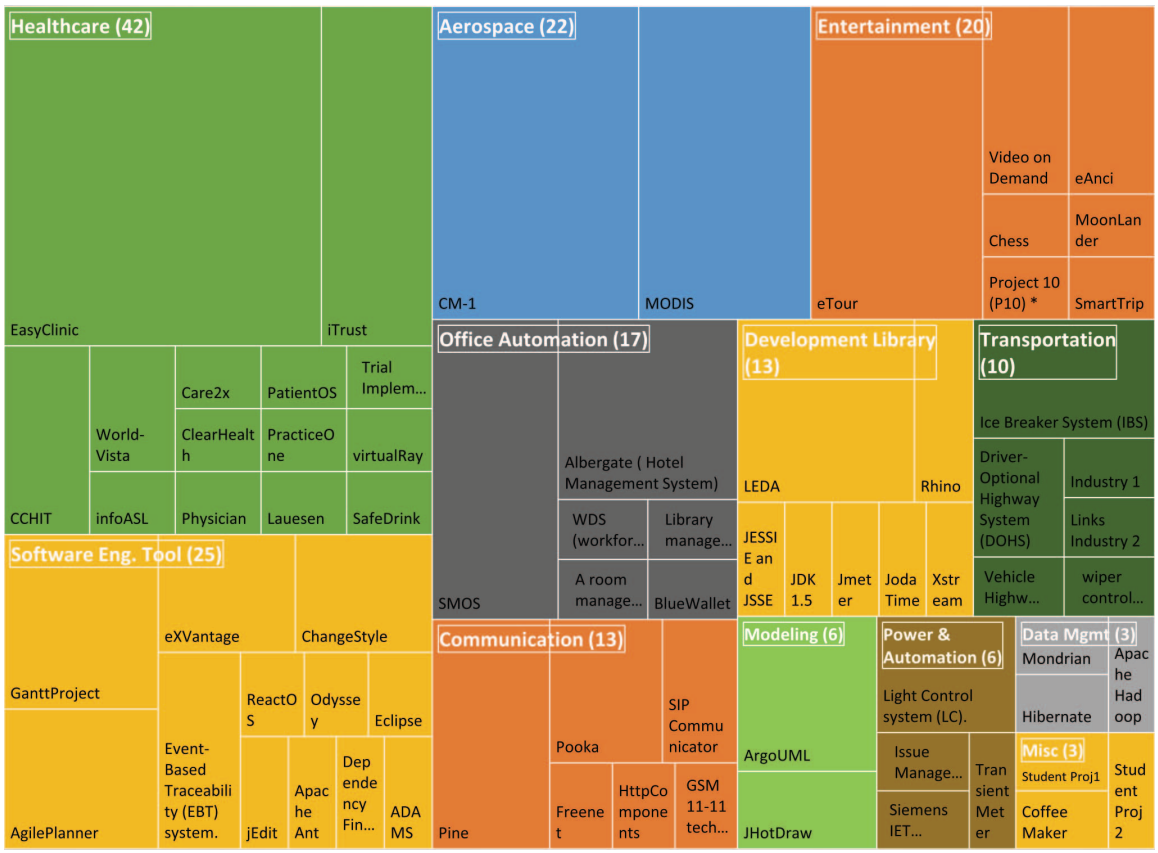

Fig. 3: Dataset Domains and Frequency of Use.

the trace space size among the three different datasets sources is relatively small.

TABLE V: Datasets' trace space statistics.

Statistics	OSS	Private/Industrial	University/Students
Minimum	264	42	50
First Quartile	870	1082	1515
Median	2028	2926	5135
Third Quartile	6956	131690	15472
Maximum	49810	29573880	390978

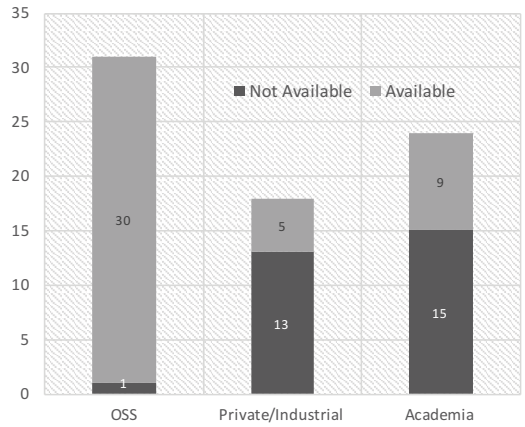

Fig. 4: Source and availability of datasets.

RQ1.4: What proportion of the traceability datasets is from industry, open-source projects, and student generated data?

As shown in Figure 4 there is a fair distribution among the different types of sources: 31 datasets are open-source software (OSS), 24 datasets come from academia (e.g. student projects), and 18 datasets are industrial projects.

RQ1.5: Are traceability datasets available for reuse?

Figure 4 shows that 39.7% of the datasets (29 out of 73) are not available. Almost all of the OSS datasets are available. The majority of the industrial datasets (13 out of 18) are not available. The majority of the datasets coming from academia are not available (15 out of 24).

B. RQ2: How to assess the quality of traceability datasets?

Data Quality Assessment frameworks are adopted in various fields of computing such as requirement engineering [84], information systems [85] [86], web linked data [22], data-warehousing [87] [88], and health-care [89], [90] to assess the quality of scientific datasets. Some of these frameworks are specific to a domain, while others are applicable to a broad range of scientific datasets [21]. We have studied several data quality frameworks and we have adapted them to the domain

TABLE VI: Traceability-Data Quality Assessment (T-DQA) Framework: dimensions, and definitions.

Dimensions	Sub - Dimensions	Definition	Metrics	Metric Type
Accessibility	**Availability**	Availability of a dataset is the extent to which data is present, obtainable and ready for use.	Dataset can be downloaded from the link provided.	Binary
	Licensing	Licensing is defined as the granting of permission for a consumer to re-use a dataset under defined conditions.	license agreement exists.	Binary
	Storage	Where is the Dataset Stored?	Private Server/ Organizational Server /Public	Categorical
Intrinsic	**Domain**	A specified sphere of activity or knowledge having common set of requirements, terminology, and functionality	Application Domain	Categorical
	Completeness	Completeness refers to the degree to which all required information is present in a particular dataset.	Source-Target artifacts are present. (at least 2 types of artifacts are present in dataset)	Binary
			Answer Set is Present	Binary
	Developers	Team responsible for creating the dataset.	Open source community, industrial or academic	Categorical
	Programming Language	A programming language is a formal computer language designed to communicate instructions to a machine.	Java/ C++ etc.	Categorical
Contextual	**Relevancy**	Relevancy refers to the provision of information which is in accordance with the task at hand and important to the users' query.	Size – Total number of artifacts	Numerical
			Number of artifacts for each specific type (Req., UML Diagrams, Code, Test, and etc.)	(Type, Numerical)
	Trustworthiness	Trustworthiness is defined as the degree to which the information is accepted to be correct, true, real, and credible.	Dataset Source	Categorical
			Frequency of Usage	Numerical
Representational	**Interpretability**	It refers to technical aspects of the data, that is, whether information is represented using an appropriate notation and whether the machine is able to process the data.	detecting the use of appropriate language, symbols, units, datatypes and clear definitions	Categorical

of software traceability. We propose a Traceability-Data Quality Assessment (T-DQA) framework that is suited for assessing datasets in the field of traceability.

Following the approach of Wang et al. [21] we broadly classified the T-DQA under four main quality dimensions: *intrinsic*, *contextual*, *representational*, and *accessibility*. These in turn comprise of 10 sub-dimensions and 13 quality metrics. Table VI provides a summary of the proposed T-DQA framework with the dimensions, sub-dimensions, definitions, metrics definitions and types. The metrics are adapted from the existing literature on assessment of data quality in a broad sense [21] and from the domain of *linked data* which to some extent is similar to traceability [22].

To illustrate the different dimensions in the T-DQA framework, we use the Easy-Clinic dataset as an example. *Easy-Clinic* implements all operations required to manage a medical ambulatory.

- The *accessibility* dimension accounts for data concerns related to access, authenticity, and retrieval [22]. The EasyClinic dataset is publicly available on the COEST GitHub repository under the General Public License.
- The *intrinsic* dimension captures the characteristics of the dataset that are inherent to itself and independent of its usage. Characteristics such as domain, completeness of dataset, developers (e.g. open source vs. industrial) and programming languages can be used as intrinsic quality indicators helping researchers to reason about the suitability of a dataset for a research problem. For instance, Easy-Clinic belongs to the HealthCare domain. It is considered complete for use in traceability as it consists of at least two artifacts (code, requirements, etc.) and it contains an answerset. It was developed by Master students at the University of Salerno. It is written in Java.
- The *contextual* dimension captures how suitable a dataset is in a particular research context. While a dataset might be good for tracing requirements to source code, it might not be suitable for tracing requirements to tests or to design documents. Contextual quality indicators are the relevancy of a dataset to a research problem and the trustworthiness of the dataset in that context. Frequently used datasets for evaluating a particular research problem are typically considered as benchmark data, which adds to the reputability of the datasets, and facilitates the comparison of the results across different papers [21]. Easy-Clinic consists of 30 use cases, 20 interaction diagrams, 63 test cases, and 37 code classes, accounting for a total of 160 artifacts and an answerset of 1005 trace links. The number of artifacts are contextual metrics as their usage is relative to the traceability task at hand. For instance, in case of requirement-to-requirement traceability, datasets having higher number of requirement artifacts such as CM1 (235 High Level X 220 Low Level) and MODIS (19 High Level X 49 Low level) are better suited than others that have lower number of requirement artifacts. Easy-Clinic is a student project. Its reuse factor is 20 as it is used as dataset in 20 papers.
- The *representational* dimension concerns the format in which a dataset is available. The quality of a dataset depends on how the dataset is packed and shared with others. This metric is indicative of the dataset formats such as XML, PDF, Word, Source-Code. EasyClinic is available on COEST website in XML format.

This framework helps researchers to be conscious of the different dimensions that need to be considered when

choosing a dataset for a traceability research problem at hand.

C. RQ3: Is there a relation between the characteristics and the quality of traceability datasets on the one hand and their reusability on the other hand?

To answer this research question, we examined the relationship between reusability of a dataset (dependent variable) and the dataset's quality metrics as depicted in column "Metrics" of Table VI (independent variables). We selected all datasets for which we were able to retrieve or calculate the values for the quality metrics, i.e., 46 datasets.

Usability is a dichotomous variable, thus to compute the values for all datasets we encoded all datasets used at least twice as 1 and all datasets that are used only once as 0. We built a Random Forest importance plot [24] to determine the metrics that best predict the reusablity of datasets.

Figure 5 indicates the importance of the quality metrics for the reusability of datasets. We observe that the most important metric is the team of *developers* that creates the dataset. This can be explained by the fact that almost all of the open source projects are available and thus facilitate reusability. *Size*, which corresponds to the number of artifacts in the dataset, appears to be an important factor as well. Other important factors are the *domain*, which is in accordance with the discussion in section RQ1.2, and the completeness of the datasets in terms of *source and target* artifacts.

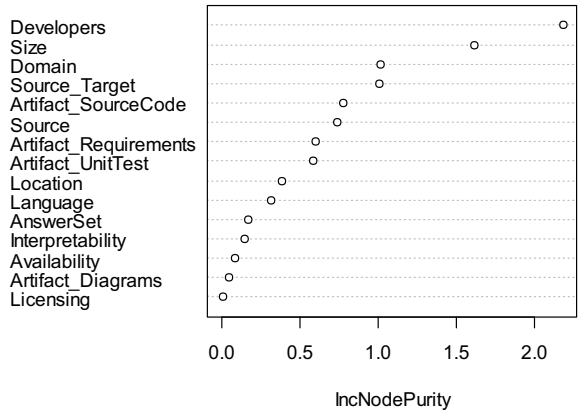

Fig. 5: Random Forest Importance Variable Plot.

Additionally, we built a linear regression model with the *actual frequency of use* as dependent variable. When performing multivariate regression we must account for possible risk of multicollinearity (i.e., interaction among the independent variables). A common way to deal with multicollinearity is to compute the Variance Inflation Factors (VIF) for each independent variable in the regression model and retain only those with low values—e.g., $\leq$2.5 [91], [92]. After removing independent variables and non-significant variables, the only remaining independent variable is AnswerSet with coefficient 6.8352 (p-value=2.67e-07). The percentage of variance of the data explained by the model is about 45%.

D. RQ4: What are the threats to validity associated with traceability datasets?

Among the 78 papers included in our SLR, 40% did not include a section related to the threats to validity nor discussed such concern while heavily relying on datasets to make research conclusion. 6% of the papers did have a threats to validity section, but did not identify any threats related to the usage of the data in their studies. Lastly, 54% of the papers discussed the threats to validity of their research related to the usage of datasets.

Two of the authors extracted all the threats to validity related to the datasets and manually grouped them. Note that we include all threats that are discussed by the authors of the respective papers which means that they were not necessarily mitigated. The threats to validity are as follows:

- *Trustworthiness*
  - *Artificial AnswerSet:* This threat is concerned with how answersets are created [4], [7], [10], [37], [45], [50], [61], [93]. Often the trustworthiness threat is not mitigated as the answersets are established by students rather than the original developers.
  - *Students Dataset*: This threat concerns dataset that are developed by students [38].
  - *Vetting Datasets*: This threat concerns datasets, particularly answersets, that are not vetted nor peer-reviewed [4], [50].
- *Threats to external validity*
  - *Real-World Data*: This threat is concerned with whether the datasets are representative of industrial projects [4], [6], [8], [28], [29], [31]–[33], [35], [38], [42]–[45], [50], [53], [56], [61], [65], [77], [94]–[96].
  - *Limited Observations*: This threat is concerned with whether a limited number of case studies are used to validate the results [4], [7], [29]–[31], [36], [37], [41], [49], [50], [52], [53], [68], [77], [95], [97].
  - *Domain*: This threat is a concern when all datasets belong to the same application domain [2], [10], [56], [61], [94], [97] or when the number of datasets is insufficient to generalize the conclusions for a particular domain.
  - *Cross Industry*: When an industrial dataset is used, this threat is concerned with whether the results are applicable to other industrial systems [98].

- *Size*: This threat is related to the small size of datasets, impacting the generalizability of the results [2], [28], [31], [33], [34], [44], [45], [50], [53], [56], [62], [65], [66], [77], [94].
- *Programming Language*: This threat is a concern when datasets are in a specific programming language [38], [47], [48].
- *Artifact Type*: This threat is concerned with the diversity of the type of artifacts available for the datasets (e.g., requirements, test cases, etc.) [2], [41].

- *Data Acquisition*
  - *Selection bias*: When datasets are not representative of the intended population (cherry picking) [6] or do not fir the problem [6], [8], [49], [75], [99]. For instance, this happens when a dataset from a non-safety critical project is used for a safety critical research study.
  - *Dataset-Equivalency*: This threat to validity concerns cases where researchers compare certain characteristics of their datasets with datasets used by previous researchers to justify the adequacy of the selected datasets [8], [31].
  - *Information bias*: Accuracy of the automatically generated datasets; misclassification and labeling of the data to be used [47], [47], [48], [75].
  - *Negative Set Bias*: Rich and unbiased selection of negative cases in training data, a common threat in classification problems [4].

E. RQ5: Do we, as a community, strive for a diversity of traceability datasets?

To answer this research question, we studied the diversity of datasets used by authors across different research papers. First, we identified all authors who have published more than one traceability paper. This took us from 128 authors served on the 78 studied papers to 38 authors who have published more than one paper. For each of these authors, we calculated the diversity metric defined in Equation 2.

Figure 6 shows the results. The X-axis represents the total number of datasets used by each author. The Y-axis represents the total number of papers from each author in this SLR. The Z-axis represents the diversity ratio for the datasets used by the authors. Each vertical dropline corresponds to one of the 38 authors. 12 authors from these 38, have a diversity ratio of 70% and above, 27 authors have a diversity rate of 50% and above, and lastly, 11 authors have a diversity rate below 50%.

We observe that in general authors with low number of datasets and low number of papers have a higher diversity ratio. One of the authors with high number of datasets and high number of papers has a high diversity rate. This example highlights individual effort in seeking diverse datasets for development and evaluation of various traceability solutions.

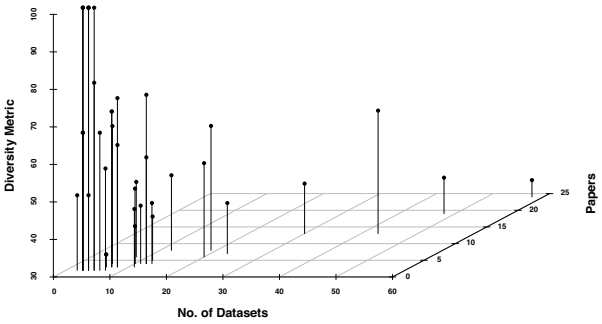

Fig. 6: Authors and their Dataset Diversity.

V. Threats To Validity

There are two main threats to validity impacting this SLR: bias in the study selection and bias in the extraction of data. Study selection can be dependent on the individuals reviewing the papers and therefore, the researchers knowledge in the domain could affect the results of this SLR. To minimize such bias, we followed a restrict protocol in which two authors of this paper selected the papers and documented their rationale for including or excluding the paper from the study. Then, the decisions were reviewed in follow-up group studies by all of the authors. The search strategy includes both automatic and manual approaches. The automatic search relies on the title, abstract, and keywords of papers. The manual search was conducted to complement the automated search and reduce the chances of missing relevant papers from known traceability venues.

To minimize the threat related to the data extraction process, we established Google Doc spreadsheets that were used to collect data from each study and to document agreements and disagreements between two individuals who collected the data. All extracted information was discussed in group meetings involving all authors. Therefore, all data points have been reviewed by all the authors to minimize biases and mistakes in collection of the data. To reduce errors while collecting information for each dataset, we used several papers that used the same datasets as a sanity check. In case a dataset was available online, we reviewed the dataset itself to complete the information.

VI. Discussion and Conclusion

What can be learned from this study? First, this study highlights the detailed characteristics of the datasets used in the domain of software traceability. This provides an in-depth understanding of the current state of the datasets used in the community, their strength, and shortcomings. Such novel knowledge draws the attention of our community to the areas that can improve rigorousness of evaluation and practicality of research.

TABLE VII: Datasets' information and the studied papers which used the datasets.

Paper	DataSet Name	General Domain	Avail	URL	OSS	Ind.	Aca.
[58]	Odyssey	SE Tool	N	NA	N	N	Y
[93]	Mondrian	Data Management	Y	https://goo.gl/uQRIRC	Y	N	N
[63], [64], [66], [93]	AgilePlanner	SE Tool	Y	https://goo.gl/nROJEo	Y	N	N
[59]	JESSIE and JSSE	Development Library	Y	https://goo.gl/gqk98Q	Y	N	N
[60]	Eclipse	SE Tool	Y	http://www.eclipse.org	Y	N	N
[60], [62]	Rhino	Development Library	Y	https://goo.gl/YemfFG	Y	N	N
[61]	JDK 1.5	Development Library	Y	http://goo.gl/mznQqF	Y	N	N
[61], [63], [64], [66]	ArgoUML	Modeling	Y	http://goo.gl/G9PzHC	Y	N	N
[61]	Freenet	Communication	Y	https://goo.gl/0I8XYi	Y	N	N
[61]	Jmeter	Development Library	Y	http://jmeter.apache.org/	Y	N	N
[5], [33], [41], [44], [45]	Pine	Communication	N	NA	N	N	Y
[31], [62], [95]	Pooka	Communication	Y	https://goo.gl/nfXrrg	Y	N	N
[62]	jEdit	SE Tool	Y	http://www.jedit.org/	Y	N	N
[68]	Joda Time	Development Library	Y	https://goo.gl/sNxXoN	Y	N	N
[68]	HttpComponents	Communication	Y	https://goo.gl/cIqNuX	Y	N	N
[68]	Hibernate	Data Management	Y	https://goo.gl/f3n52Z	Y	N	N
[68]	Xstream	Development Library	Y	https://goo.gl/e447m6	Y	N	N
[16], [50], [65], [75], [76]	GanttProject	SE Tool	Y	https://goo.gl/MlJx4D	Y	N	N
[75], [76]	JHotDraw	Modeling	Y	https://goo.gl/sNzUkb/	Y	N	N
[63], [64], [66]	eXVantage	SE Tool	N	NA	N	Y	N
[15], [41], [69]	ChangeStyle	SE Tool	N	NA	N	N	Y
[65]	ReactOS	SE Tool	Y	https://goo.gl/4xv4Qc	Y	N	N
[52], [66], [67]	EBT system.	SE Tool	N	NA	N	N	Y
[52]	SE450 Projects	Miscellenous	N	NA	N	N	Y
[64]	Apache Ant	SE Tool	Y	https://goo.gl/uJC8wx	Y	N	N
[64]	Dependency Finder	SE Tool	Y	https://goo.gl/fgE1su	Y	N	N
[9]	ADAMS	SE Tool	N	NA	N	N	Y
[4]	Apache Hadoop	Data Management	Y	https://goo.gl/zLI9ZW	Y	N	N
[9], [70]–[73], [73]	LEDA	Development Library	Y	http://goo.gl/RbyyoM	N	N	Y
[2], [6], [7]	CCHIT	Healthcare	Y	https://goo.gl/3KPkmO	Y	N	N
[2], [6]	World- Vista	Healthcare	Y	https://goo.gl/Sv8DDK	Y	N	N
[28]	infoASL	Healthcare	N	NA	N	N	Y
[6], [7], [29]–[33]	iTrust	Healthcare	Y	https://goo.gl/Jz7HQn	N	N	Y
[6]	Care2x	Healthcare	Y	https://goo.gl/2VIdRL	Y	N	N
[6]	ClearHealth	Healthcare	Y	https://goo.gl/xLS2qM	Y	N	N
[6]	Physician	Healthcare	N	NA	N	Y	N
[6]	PatientOS	Healthcare	Y	https://goo.gl/u71qPZ	Y	N	N
[6]	Trial Implementations	Healthcare	Y	https://goo.gl/vLHwgR	N	Y	N
[6]	PracticeOne	Healthcare	Y	https://goo.gl/W9sH3q	N	Y	N
[6]	Lauesen	Healthcare	Y	https://goo.gl/7Hblkc	N	N	Y
[28]	virtualRay	Healthcare	N	NA	N	N	Y
[16]	WV-CCHIT	Healthcare	Y	https://goo.gl/hbwgsz	Y	N	N
[34]	SafeDrink	Healthcare	N	NA	N	Y	N
[7], [9], [16], [28], [30], [33], [35]–[48]	EasyClinic	Healthcare	Y	https://goo.gl/1RxkzC	N	N	Y
[5], [7], [11], [15], [16], [32], [41], [44], [49]–[52]	CM-1	Aerospace	Y	https://goo.gl/K1GwSh	N	Y	N
[15], [30], [33], [39], [44], [51], [53]–[56]	MODIS	Aerospace	Y	https://goo.gl/AzOnSm	N	Y	N
[76]	Chess	Entertainment	N	NA	Y	N	N
[16], [71]–[73], [73], [79]	Albergate	Office Automation	Y	https://goo.gl/Vc0NGT	N	N	Y
[94]	CoffeeMaker	Miscellenous	Y	https://goo.gl/mzoTtr	N	N	Y
[74]–[76]	Video on Demand	Entertainment	Y	https://goo.gl/oEZPsW/	Y	N	N
[3]	Project 10 (P10) *	Entertainment	Y	https://goo.gl/dRQwRt	Y	N	N
[56]	MoonLander	Entertainment	N	NA	N	N	Y
[1], [52], [67], [83]	Ice Breaker System	Transportation	N	NA	N	N	Y
[27]	Vehicle Highway System	Transportation	N	NA	N	Y	N
[10], [97]	Driver-Optional Highway System (DOHS)	Transportation	N	NA	N	Y	N
[7]	Industry 1	Transportation	N	NA	N	Y	N
[7]	Links Industry 2	Transportation	N	NA	N	Y	N
[78]	Wiper control system	Transportation	N	NA	N	Y	N
[62], [95]	SIP Communicator	Communication	Y	https://goo.gl/CulKDb	Y	N	N
[99]	GSM 11-11	Communication	Y	https://goo.gl/7BsrCW	N	Y	N
[80]	Issue Management System (IMS)	power and automation	N	NA	N	Y	N
[81]	Siemens IET document	power and automation	N	NA	N	Y	N
[82]	Transient Meter	power and automation	N	NA	N	N	Y
[98]	Workforce development	Office Automation	N	NA	N	Y	N
[34]	BlueWallet	Others	N	NA	N	Y	N
[52], [67], [83]	Light Control system	power and automation	N	NA	N	N	Y
[1], [8], [15], [29], [32], [33], [38], [42], [44], [79]	eTour	Entertainment	Y	https://goo.gl/bYLzQD	N	N	Y
[77]	Room management	Office Automation	N	NA	N	N	Y
[34]	SmartTrip	Entertainment	N	NA	N	Y	N
[16], [28], [38]	eAnci	Entertainment	Y	https://goo.gl/9sVdFu	N	N	Y
[78]	Library system	Office Automation	N	NA	N	N	Y
[8], [16], [28], [38], [42], [43], [79]	SMOS	Office Automation	Y	https://goo.gl/UALgaf	N	N	Y
[57]	7 Student Projects	Miscellenous	N	NA	N	N	Y

Avail: If dataset is available; OSS: Open Source Project; Ind.: Industrial Project; Aca.: Academic Project

We also adapted existing quality assessment frameworks to the domain of software traceability to help researchers evaluate the quality of the datasets and their relevancy for specific research tasks. T-DQA is the first proposed solution in the community and it relies on the knowledge and frameworks used in other areas of computing. We have collected 73 datasets, evaluated them using the T-DQA framework, and released the results publicly. These results can be used as guidelines for the researchers to select datasets based on their research needs and the characteristics of the datasets. For instance, for each of these datasets (reported in Table VII), we provide a link to the source, we describe the meta-data associated with it, previous studies that used it, threats to validity associated with it, trace space, and other characteristics of the dataset. Using such extensive guidelines, researchers can use our online material (released in github) to select an appropriate dataset based on application domain, size, or other relevant factors.

Furthermore, our study makes the tacit community wide threats related to the datasets explicit. This will help us as a community to better understand the strengths and weaknesses of our empirical foundations, but also, it will help is to make more informed decisions in assessing and improving the quality of our datasets.

REFERENCES

[1] J. Cleland-Huang, R. Settimi, O. BenKhadra, E. Berezhanskaya, and S. Christina, "Goal-centric traceability for managing non-functional requirements," in *ICSE*, 2005, pp. 362–371.

[2] M. Rahimi, M. Mirakhorli, and J. Cleland-Huang, "Automated extraction and visualization of quality concerns from requirements specifications," in *RE*, 2014, pp. 253–262.

[3] D. Port, A. Nikora, J. H. Hayes, and L. Huang, "Text mining support for software requirements: Traceability assurance," in *Hawaii Int. Conf. on System Sciences*, 2011, pp. 1–11.

[4] M. Mirakhorli and J. Cleland-Huang, "Detecting, tracing, and monitoring architectural tactics in code," *TSE*, vol. 42, no. 3, pp. 205–220, March 2016.

[5] H. Sultanov and J. Hayes, "Application of reinforcement learning to requirements engineering: requirements tracing," in *RE*, 2013, pp. 52–61.

[6] J. Cleland-Huang, A. Czauderna, M. Gibiec, and J. Emenecker, "A machine learning approach for tracing regulatory codes to product specific requirements," in *ICSE - Volume 1*, 2010, pp. 155–164.

[7] S. Lohar, S. Amornborvornwong, A. Zisman, and J. Cleland-Huang, "Improving trace accuracy through data-driven configuration and composition of tracing features," in *ESEC/FSE*. ACM, 2013, pp. 378–388.

[8] D. Diaz, G. Bavota, A. Marcus, R. Oliveto, S. Takahashi, and A. De Lucia, "Using code ownership to improve ir-based traceability link recovery," in *ICPC*, 2013, pp. 123–132.

[9] A. De Lucia, F. Fasano, R. Oliveto, and G. Tortora, "Can information retrieval techniques effectively support traceability link recovery?" in *ICPC*, 2006, pp. 307–316.

[10] J. Guo, N. Monaikul, C. Plepel, and J. Cleland-Huang, "Towards an intelligent domain-specific traceability solution," in *ASE*, 2014, pp. 755–766.

[11] S. Yadla, H. J. Hayes, and A. Dekhtyar, "Tracing requirements to defect reports: an application of information retrieval techniques," *ISSE*, vol. 1, no. 2, pp. 116–124, 2005.

[12] G. A. Liebchen and M. Shepperd, "Data sets and data quality in software engineering," in *Int. Workshop on Predictor Models in Software Engineering (PROMISE)*, 2008, pp. 39–44.

[13] M. Shepperd, Q. Song, Z. Sun, and C. Mair, "Data quality: Some comments on the nasa software defect datasets," *TSE*, vol. 39, no. 9, pp. 1208–1215, Sept 2013.

[14] J. Cleland-Huang, O. C. Gotel, J. Huffman Hayes, P. Mäder, and A. Zisman, "Software traceability: trends and future directions," in *Future of Software Engineering*, 2014, pp. 55–69.

[15] J. H. Hayes, G. Antoniol, B. Adams, and Y. G. Guhneuc, "Inherent characteristics of traceability artifacts less is more," in *RE*, 2015, pp. 196–201.

[16] Y. Shin, J. Hayes, and J. Cleland-Huang, "Guidelines for benchmarking automated software traceability techniques," in *SST*, 2015, pp. 61–67.

[17] M. Borg, P. Runeson, and A. Ardö, "Recovering from a decade: a systematic mapping of information retrieval approaches to software traceability," *EMSE*, vol. 19, no. 6, pp. 1565–1616, 2014.

[18] S. Nair, J. L. De La Vara, and S. Sen, "A review of traceability research at the requirements engineering conference re@ 21," in *RE*, 2013, pp. 222–229.

[19] I. Santiago, A. Jiménez, J. M. Vara, V. De Castro, V. A. Bollati, and 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.

[20] M. F. Bosu and S. G. MacDonell, "A taxonomy of data quality challenges in empirical software engineering," in *Australian Software Engineering Conf. (ASWEC)*, 2013, pp. 97–106.

[21] R. Y. Wang and D. M. Strong, "Beyond accuracy: What data quality means to data consumers," *Journal of Management Information Systems*, vol. 12, no. 4, pp. 5–33, 1996.

[22] A. Zaveri, A. Rula, A. Maurino, R. Pietrobon, J. Lehmann, and S. Auer, "Quality assessment for linked data: A survey," *Semantic Web*, vol. 7, no. 1, pp. 63–93, 2016.

[23] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," EBSE Technical Report EBSE-2007-01, Tech. Rep., 2007.

[24] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[25] H. Zhang, M. A. Babar, and P. Tell, "Identifying relevant studies in software engineering," *Information and Software Technology*, vol. 53, no. 6, pp. 625–637, 2011.

[26] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," *JSS*, vol. 80, no. 4, pp. 571 – 583, 2007.

[27] Y. Li and J. Cleland-Huang, "Ontology-based trace retrieval," in *TEFSE*, 2013, pp. 30–36.

[28] G. Bavota, A. De Lucia, R. Oliveto, A. Panichella, F. Ricci, and G. Tortora, "The role of artefact corpus in lsi-based traceability recovery," in *TEFSE*, 2013, pp. 83–89.

[29] A. Mahmoud and N. Niu, "Source code indexing for automated tracing," in *TEFSE*, 2011, pp. 3–9.

[30] A. Panichella, A. De Lucia, and A. Zaidman, "Adaptive user feedback for ir-based traceability recovery," in *SST*, 2015, pp. 15–21.

[31] N. Ali, Z. Sharafl, Y. Gueheneuc, and G. Antoniol, "An empirical study on requirements traceability using eye-tracking," in *ICSM*, 2012, pp. 191–200.

[32] N. Niu and A. Mahmoud, "Enhancing candidate link generation for requirements tracing: The cluster hypothesis revisited," in *RE*, 2012, pp. 81–90.

[33] A. D. Lucia, M. D. Penta, R. Oliveto, A. Panichella, and S. Panichella, "Applying a smoothing filter to improve ir-based traceability recovery processes: An empirical investigation," *Information and Software Technology*, vol. 55, no. 4, pp. 741–754, 2013.

[34] A. Mahmoud, "An information theoretic approach for extracting and tracing non-functional requirements," in *RE*, 2015, pp. 36–45.

[35] A. De Lucia, R. Oliveto, and G. Tortora, "Assessing ir-based traceability recovery tools through controlled experiments," *EMSE*, vol. 14, no. 1, pp. 57–92, 2009.

[36] ——, "IR-based traceability recovery processes: An empirical comparison of "one-shot" and incremental processes," in *ASE*, 2008, pp. 39–48.

[37] A. D. Lucia, F. Fasano, R. Oliveto, and G. Tortora, "Recovering traceability links in software artifact management systems using information retrieval methods," *TOSEM*, vol. 16, no. 4, Sep. 2007.

[38] M. Gethers, R. Oliveto, D. Poshyvanyk, and A. Lucia, "On integrating orthogonal information retrieval methods to improve traceability recovery," in *ICSM*, 2011, pp. 133–142.

[39] A. De Lucia, R. Oliveto, and P. Sgueglia, "Incremental approach and user feedbacks: a silver bullet for traceability recovery," in *ICSM*, 2006, pp. 299–309.

[40] A. De Lucia, F. Fasano, R. Oliveto, and G. Tortora, "Enhancing an artefact management system with traceability recovery features," in *ICSM*, 2004, pp. 306–315.

[41] W. K. Kong and J. H. Hayes, "Proximity-based traceability: An empirical validation using ranked retrieval and set-based measures," in *Workshop on Empirical Requirements Engineering (EmpiRE)*, 2011, pp. 45–52.

[42] A. Panichella, C. McMillan, E. Moritz, D. Palmieri, R. Oliveto, D. Poshyvanyk, and A. De Lucia, "When and how using structural information to improve ir-based traceability recovery," in *CSMR*, 2013, pp. 199–208.

[43] G. Bavota, A. D. Lucia, R. Oliveto, and G. Tortora, "Enhancing software artefact traceability recovery processes with link count information," *Information and Software Technology*, vol. 56, no. 2, pp. 163–182, 2014.

[44] G. Capobianco, A. De Lucia, R. Oliveto, A. Panichella, and S. Panichella, "Improving ir-based traceability recovery via noun-based indexing of software artifacts," *SEP*, vol. 25, no. 7, pp. 743–762, 2013.

[45] A. De Lucia, M. Di Penta, R. Oliveto, A. Panichella, and S. Panichella, "Improving ir-based traceability recovery using smoothing filters," in *ICPC*, 2011, pp. 21–30.

[46] A. De Lucia, R. Oliveto, and G. Tortora, "The role of the coverage analysis during ir-based traceability recovery: A controlled experiment," in *ICSM*, 2009, pp. 371–380.

[47] G. Capobianco, A. De Lucia, R. Oliveto, A. Panichella, and S. Panichella, "On the role of the nouns in ir-based traceability recovery," in *ICPC*, 2009, pp. 148–157.

[48] ——, "Traceability recovery using numerical analysis," in *WCRE*, Oct 2009, pp. 195–204.

[49] V. Gervasi and D. Zowghi, "Supporting traceability through affinity mining," in *RE*, 2014, pp. 143–152.

[50] E. Holbrook, J. Hayes, and A. Dekhtyar, "Toward automating requirements satisfaction assessment," in *RE*, 2009, pp. 149–158.

[51] S. K. Sundaram, J. H. Hayes, and A. Dekhtyar, "Baselines in requirements tracing," in *Workshop on Predictor Models in Software Engineering (PROMISE)*, 2005, pp. 1–6.

[52] X. Zou, R. Settimi, and J. Cleland-Huang, "Improving automated requirements trace retrieval: a study of term-based enhancement methods," *EMSE*, vol. 15, no. 2, pp. 119–146, 2009.

[53] S. Pandanaboyana, S. Sridharan, J. Yannelli, and J. Hayes, "Requirements tracing on target (retro) enhanced with an automated thesaurus builder: An empirical study," in *TEFSE*, 2013.

[54] J. Hayes, A. Dekhtyar, S. Sundaram, and S. Howard, "Helping analysts trace requirements: an objective look," in *RE*, 2004, pp. 249–259.

[55] J. Hayes, A. Dekhtyar, and J. Osborne, "Improving requirements tracing via information retrieval," in *RE*, 2003, pp. 138–147.

[56] W. Li, J. Hayes, F. Yang, K. Imai, J. Yannelli, C. Carnes, and M. Doyle, "Trace matrix analyzer (tma)," in *TEFSE*, 2013, pp. 44–50.

[57] A. De Lucia, F. Fasano, R. Oliveto, and G. Tortora, "Adams retrace: A traceability recovery tool," in *CSMR*, 2005, pp. 32–41.

[58] L. Murta, A. Van Der Hoek, and C. Werner, "Archtrace: Policy-based support for managing evolving architecture-to-implementation traceability links," in *ASE*, 2006, pp. 135–144.

[59] Y. Yu, J. Jurjens, and J. Mylopoulos, "Traceability for the maintenance of secure software," in *ICSM*, 2008, pp. 297–306.

[60] C. S. Corley, N. A. Kraft, L. H. Etzkorn, and S. K. Lukins, "Recovering traceability links between source code and fixed bugs via patch analysis," in *TEFSE*, 2011, pp. 31–37.

[61] X. Chen and J. Grundy, "Improving automated documentation to code traceability by combining retrieval techniques," in *ASE*, 2011, pp. 223–232.

[62] N. Ali, Y. Gueheneuc, and G. Antoniol, "Trustrace: Mining software repositories to improve the accuracy of requirement traceability links," *TSE*, vol. 39, no. 5, pp. 725–741, May 2013.

[63] A. Qusef, G. Bavota, R. Oliveto, A. De Lucia, and D. Binkley, "Scotch: Test-to-code traceability using slicing and conceptual coupling," in *ICSM*, 2011, pp. 63–72.

[64] A. Qusef, G. Bavota, R. Oliveto, A. D. Lucia, and D. Binkley, "Recovering test-to-code traceability using slicing and textual analysis," *JSS*, vol. 88, pp. 147–168, 2014.

[65] A. Egyed, F. Graf, and P. Grunbacher, "Effort and quality of recovering requirements-to-code traces: Two exploratory experiments," in *RE*, 2010, pp. 221–230.

[66] A. Qusef, G. Bavota, R. Oliveto, A. De Lucia, and D. Binkley, "Evaluating test-to-code traceability recovery methods through controlled experiments," *SEP*, no. 11, pp. 1167–1191, 2013.

[67] Z. Xuchang, R. Settimi, and J. Cleland-Huang, "Phrasing in dynamic requirements trace retrieval," in *COMPSAC*, vol. 1, 2006, pp. 265–272.

[68] B. Dagenais and M. P. Robillard, "Recovering traceability links between an api and its learning resources," in *ICSE*, 2012.

[69] W.-K. Kong, J. Huffman Hayes, A. Dekhtyar, and J. Holden, "How do we trace requirements: An initial study of analyst behavior in trace validation tasks," in *Int. Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*, 2011, pp. 32–39.

[70] H. yi Jiang, T. Nguyen, I.-X. Chen, H. Jaygarl, and C. Chang, "Incremental latent semantic indexing for automatic traceability link evolution management," in *ASE*, 2008, pp. 59–68.

[71] G. Antoniol, G. Canfora, G. Casazza, and A. De Lucia, "Information retrieval models for recovering traceability links between code and documentation," in *ICSM*, 2000, pp. 40–49.

[72] A. Marcus and J. I. Maletic, "Recovering documentation-to-source-code traceability links using latent semantic indexing," in *ICSE*, 2003, pp. 125–135.

[73] G. Antoniol, G. Canfora, G. Casazza, A. De Lucia, and E. Merlo, "Recovering traceability links between code and documentation," *TSE*, vol. 28, no. 10, pp. 970–983, Oct 2002.

[74] A. Egyed and P. Grunbacher, "Automating requirements traceability: Beyond the record replay paradigm," in *ASE*, 2002, pp. 163–171.

[75] H. Kuang, P. Mader, H. Hu, A. Ghabi, L. Huang, L. Jian, and A. Egyed, "Do data dependencies in source code complement call dependencies for understanding requirements traceability?" in *ICSM*, 2012, pp. 181–190.

[76] A. Ghabi and A. Egyed, "Code patterns for automatically validating requirements-to-code traces," in *ASE*, 2012, pp. 200–209.

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

[78] P. Mader, O. Gotel, and I. Philippow, "Rule-based maintenance of post-requirements traceability relations," in *RE*, 2008, pp. 23–32.

[79] T. Dasgupta, M. Grechanik, E. Moritz, B. Dit, and D. Poshyvanyk, "Enhancing software traceability by automatically expanding corpora with relevant documentation," in *ICSM*, 2013, pp. 320–329.

[80] M. Borg, O. Gotel, and K. Wnuk, "Enabling traceability reuse for impact analyses: A feasibility study in a safety context," in *TEFSE*, 2013, pp. 72–78.

[81] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, "The detection and classification of non-functional requirements with application to early aspects," in *RE*, 2006, pp. 39–48.

[82] M. Di Penta, S. Gradara, and G. Antoniol, "Traceability recovery in rad software systems," in *Int. Workshop on Program Comprehension (IWPC)*, 2002, pp. 207–216.

[83] J. Cleland-Huang, R. Settimi, C. Duan, and X. Zou, "Utilizing supporting evidence to improve dynamic requirements traceability," in *RE*, 2005, pp. 135–144.

[84] A. Davis, S. Overmyer, K. Jordan, J. Caruso, F. Dandashi, A. Dinh, G. Kincaid, G. Ledeboer, P. Reynolds, P. Sitaram *et al.*, "Identifying and measuring quality in a software requirements specification," in *Int. Soft. Metrics Symp.*, 1993, pp. 141–152.

[85] R. Kovac, Y. W. Lee, and L. Pipino, "Total data quality management: The case of iri." in *Conf. on Information Quality*, 1997, pp. 63–79.

[86] V. V. Mandke and M. K. Nayar, "Information integrity: A structure for its definition," in *Conf. on Information Quality*, 1997, pp. 314–338.

[87] M. Jarke and Y. Vassiliou, "Data warehouse quality: A review of the dwq project." in *IQ*, 1997, pp. 299–313.

[88] A. Matsumura and N. Shouraboura, "Competing with quality information." in *IQ*, 1996, pp. 72–86.

[89] J. ODonoghue, T. OKane, J. Gallagher, G. Courtney, A. Aftab, A. Casey, J. Torres, and P. Angove, "Modified early warning scorecard: the role of data/information quality within the decision making process," *Electronic Journal Information Systems Evaluation Volume*, vol. 14, no. 1, 2011.

[90] M. G. Kahn, M. A. Raebel, J. M. Glanz, K. Riedlinger, and J. F. Steiner, "A pragmatic framework for single-site and multisite data quality assessment in electronic health record-based clinical research," *Medical care*, vol. 50, 2012.

[91] M. Cataldo, A. Mockus, J. Roberts, and J. Herbsleb, "Software dependencies, work dependencies, and their impact on failures," *TSE*, vol. 35, no. 6, pp. 864–878, November 2009.

[92] E. Shihab, Z. M. Jiang, W. M. Ibrahim, B. Adams, and A. E. Hassan, "Understanding the impact of code and process metrics on post-release defects: A case study on the Eclipse project," in *Int. Symp. on Empirical Software Engineering and Measurement (ESEM)*, 2010, pp. 4:1–4:10.

[93] A. Qusef, R. Oliveto, and A. De Lucia, "Recovering traceability links between unit tests and classes under test: An improved method," in *ICSM*, 2010, pp. 1–10.

[94] C. McMillan, D. Poshyvanyk, and M. Revelle, "Combining textual and structural analysis of software artifacts for traceability link recovery," in *TEFSE*, 2009, pp. 41–48.

[95] N. Ali, Y. Gueheneuc, and G. Antoniol, "Trust-based requirements traceability," in *ICPC*, June 2011, pp. 111–120.

[96] A. Mahmoud and N. Niu, "Supporting requirements to code traceability through refactoring," *REJ*, vol. 19, no. 3, pp. 309–329, 2013.

[97] J. Guo, J. Cleland-Huang, and B. Berenbach, "Foundations for an expert system in domain-specific traceability," in *RE*, 2013, pp. 42–51.

[98] N. Niu, T. Bhowmik, H. Liu, and Z. Niu, "Traceability-enabled refactoring for managing just-in-time requirements," in *RE*, 2014, pp. 133–142.

[99] F. Bouquet, E. Jaffuel, B. Legeard, F. Peureux, and M. Utting, "Requirements traceability in automated test generation: Application to smart card software validation," *SIGSOFT Softw. Eng. Notes*, vol. 30, no. 4, pp. 1–7, May 2005.