Blockchain-oriented Requirements Engineering: A Framework

Selina Demi
Østfold University College
Halden, Norway
selina.demi@hiof.no

***Abstract*—This paper presents a PhD research plan that focuses on solving existing problems in requirements engineering, by means of a novel blockchain-enabled framework. Requirements engineering in general and requirements traceability in particular are characterized by a variety of distributed stakeholders and heterogeneity of tools. These characteristics introduce integration, communication, coordination and trust issues and impede the accurate and trustworthy traceability of requirements. Thereby, this paper proposes blockchain technology for the trustworthy management and traceability of requirements throughout the software development life cycle.**

Keywords-requirements engineering; requirements traceability; blockchain technology; smart contracts

I. MOTIVATION

The ever-changing technology in a highly competitive environment exerts pressure on the software development process [1]. Requirements engineering (RE) is of paramount importance in this process, however in the era of digital transformations, it has been reported that requirements engineers are not properly aiding to the development of digital solutions [2]. In particular, the reason behind this lies in the complexity of handling and achieving consensus among heterogeneous stakeholders that may not have prior knowledge about RE techniques. In fact, a recent Naming the Pain in Requirements Engineering (NaPIRE) survey positioned communication problems in the top 5 rated RE problems [3].

One of the main activities of RE is requirements management. Further, requirements traceability lies at the heart of requirements management [4], and it is mandated by standards, in particular in safety-critical systems. Surprisingly, 32.4% (158 out of 488) of the respondents in the aforementioned NaPIRE survey indicated "missing traceability" as problematic. In fact, the requirements traceability (RT) problem was identified since 1994 [5], yet to date it remains unsolved. Traceability in the software development life cycle (SDLC) is indeed important to check if requirements are properly defined, if requirements are broken down appropriately and to keep track of changes [6]. Although researchers acknowledge RT benefits, it seems that these benefits are not perceived in industry and traceability practices are not mature generally speaking [7]. To date, traceability is not adopted at all or adopted only because standards mandate it [8].

II. RELATED WORK

The first section sheds light onto existing tools and technologies used to automate RT, followed by challenges related to tools, technologies and the underlying tracing process. The last section introduces blockchain technology, its main concepts and properties and the interrelation between blockchain technology and software engineering.

A. Requirements Traceability Process, Technologies and Tools

Over years, researchers have proposed a vast amount of tools and technologies that aim to automate RT. However, full automation is still not achieved, due to the lack of trust in the existing tools, technologies and the overall RT process. In such a case, human intervention is required in order to vet the preliminary trace links generated by these tools, enabling in this way the so-called semi-automated RT. The majority of these approaches rely on information retrieval techniques (IR), because of the textual nature of requirements documents [9]. Amongst IR-based models, the most popular are vector space model [10] and latent semantic indexing [11]. However, according to [12], IR-based approaches in general have demonstrated low precision (20-50%). Therefore, attempts to improve these techniques have emerged. In [13], authors proposed a feature-oriented tool (FORTA) to automatically trace requirements to tests. The results of the evaluation revealed higher precision and recall levels compared to the aforementioned IR techniques. In [14], the tool RETRO was enhanced with a dynamic thesaurus, while in [15], refactoring was proposed to support RT.

Recently, Wang et al. [16] proposed the inclusion of term-based relevance feedback into the creation of trace links. Despite the promising results generated by their evaluations, they considered only dependability requirements. Another recent approach is to generate and maintain tracing links by means of machine learning. For instance, Mills et al. [17] proposed TRAIL and evaluated it on 11 datasets from 6 software systems. Results demonstrated better performance compared to IR-based approaches, however this approach does not take into consideration vocabulary mismatches. Shahid et al. [18] evaluated 11 requirements management and traceability tools and found out that requirements management tools such as DOORS are more popular than purely requirements traceability tools. However, none of the tools supported automatic link detection of requirements and only 5 of them guaranteed the integration with other tools throughout the

2332-6441/20/$31.00 ©2020 IEEE
DOI 10.1109/RE48521.2020.00063

software life cycle. These limitations and others (See Section B) lead to low levels of adoption of RT in practice.

B. Requirements Traceability Challenges

In 1994, Gotel and Finkelstein [5] identified and analyzed the requirements traceability problem, by means of an extensive empirical study. According to their analysis, RT problem is the umbrella under which many other problems exist, and interestingly further improvements to this field introduce further problems. Therefore, an all-encompassing solution seems infeasible [5]. The importance of RT, along with its practical limitations motivated researchers to focus on barriers that hinder the use of RT in practice, in order to provide a comprehensive understanding of the "milestones" of this problem. In 1998, Ramesh [19] published a survey conducted in 26 organizations and identified technological, organizational and environmental factors influencing the adoption of RT, just to mention a few: inadequate strategies, incentives, training and incompatibility among tools. Interestingly, these factors were interpreted by comparing two extremes of practice, namely, low-end and high-end traceability users [19].

In 2007, Blaauboer et al. [20] carried out a case study in a large IT company, to identify factors influencing the decision to adopt traceability. They found out significant factors such as: development organization awareness, customer awareness, return on investment, stakeholder preferences and process flow. Interestingly, they revealed that the majority of software development leaders were unaware about traceability, let alone using it. In 2009, Kannenberg and Saiedian [21] identified challenges that RT faces in practice. These challenges include but are not limited to, costs, difficulty of maintaining traceability through change, differences in stakeholders' viewpoints on traceability and poor tool support. In 2010, Winkler and von Pilgrim [7] published an extensive literature review on traceability research and practice in requirements engineering and model-driven development. Moreover, they identified unsolved challenges with regards to traceability in practice, categorized into natural, technical, economical and social factors, as well as differences between the aforementioned areas.

Recently, Wang et al. [9] carried out an extensive literature review focused on RT technologies, followed by a mapping of these technologies to their related challenges. In particular, they revealed that the most researched challenges are "trustworthy" and "automated", whereas challenges such as "scalable", "coordinated", "dynamic" and "lightweight" received less focus. Furthermore, Maro et al. [8] conducted a study that relies on a triangulation of data sources, more specifically, a tertiary study to identify the general challenges and solutions of RT, a multivocal literature review to identify specific challenges and solutions in the automotive domain and a case study to validate them in practice. All of these data sources reported challenges such as "diverse artifacts and tools" and "manual work" and solutions such as "tool integration" and "automation". Although an overlap between general challenges and solutions, and those in the automotive domain exists, some solutions are not feasible in the latter case. In fact, the automotive domain is characterized by distributed development, safety-critical systems and system complexity. Despite the importance of traceability, in particular in safety-critical systems, tracing activities are still perceived as an overhead and the exchange of tracing links is not yet standardized.

C. Blockchain Technology and Software Engineering

Blockchain technology is a distributed immutable ledger, which stores transactions in blocks [22]. Blocks' headers contain the hash of the previous block, forming in this way a chain of blocks. The strength of blockchain lies in the decentralized environment, where no third party is in control of data or transactions. Moreover, it can be used to create smart contracts, which are essentially programs that live on the blockchain and can express triggers, conditions and business logic [23]. Although blockchain technology has the potential to disrupt traditional business models in many areas such as voting, electronic health records and supply chain [24], surprisingly the interrelation between blockchain and software engineering has received little attention and only recently.

Blockchain enabled new forms of distributed software, which introduce many challenges such as the need for new professional roles, security and reliability, need for specialized modeling languages and for specific metrics [25]. The need to adapt standard software engineering techniques for blockchain-oriented software introduced the novel field of blockchain-oriented software engineering [25]. This is further confirmed by the survey conducted in [26], where authors concluded that traditional software engineering practices do not consider the decentralized nature of blockchain, thus there is a need for further improvement in this dimension.

The other facet of blockchain-oriented software engineering, i.e. the use of blockchain technology to improve challenges of software engineering has been less studied. Although, few use cases have been mentioned in literature, most of them have not yet been detailed or empirically validated [27]. Beller and Hejderup [28] propose blockchain-based continuous integration system, which is a decentralized system to perform build tasks and verifiable by everyone, as opposed to the centralized system of Travis continuous integration. Moreover, they introduce the concept of blockchain-based package repository, where everyone can propose the inclusion of new packages and verify others' work. Krol et al. [29] go further by presenting and evaluating smart contracts through ChainSoft, a platform for collaborative software development. Yilmaz et al. [27] aim to address the non-functional requirement of integrity of large-scale agile software development practices. Interestingly, they perceive software development as the Byzantine Generals Problem, with byzantine participants being developers who create software bugs. Therefore, they propose a test-driven incentive mechanism that makes use of blockchain principles, in which analogically software developers are miners that develop code, whereas testers are the validators who would create the block by incrementing a working piece of the software. Lenarduzzi et al. [30] suggested a blockchain-based approach for the management of agile projects based on Scrum and Lean-Kanban process. Their goal is to shift the responsibility for certifying the accuracy of the process from the product owner to smart contracts that in particular will automate the acceptance phase and the payments to developers. Bose et al. [31] propose a blockchain-enabled framework that intends to enhance the trustworthiness of

software provenance data. Conversely, this thesis aims to propose and evaluate a blockchain-enabled framework that manages and traces requirements (forward and backward traceability) throughout the software development life cycle. None of the aforementioned approaches explores the potential of blockchain in the field of RE, in particular in RT.

III. Problem and Objective

In 2011, Gotel et al. [32] published a technical report as an ongoing effort by traceability researchers and practitioners who created the Center of Excellence for Software Traceability (CoEST). They identified eight challenges of traceability, i.e. purposed, cost-effective, configurable, trusted, scalable, portable, valued and ubiquitous. The last challenge is considered the grand challenge of traceability because achieving ubiquitous traceability requires the improvement of all the other challenges. To date, these challenges still exist, leading to low adoption rates in practice [8], [33], [34]. Motivated by these facts and acknowledging the importance of traceability in a variety of aspects of software development and beyond, we carried out a rigorous systematic literature review (SLR) that intends to identify factors that influence this low adoption, hoping that the comprehensive analysis of these factors will lead us to a holistic solution. Based on our preliminary study, the most significant factors entail the lack of trust in existing tracing tools due to their low accuracy [12], inadequate integration or interoperability among heterogeneous tools [35], lack of support for change notification and propagation [36], invisible benefits [37], lack of organizational strategies and common practices throughout the organization [38], communication and coordination problems among distributed stakeholders [36] and confidentiality constraints that impede the sharing of artifacts with external organizations [8].

In fact, the idea of adopting a blockchain-oriented approach was derived from these challenges. We argue that a blockchain-oriented solution will facilitate the communication and collaboration between distributed stakeholders, will enable the reliable sharing of artifacts, incentivize practitioners to create and validate trace links and automate the compliance to standards, especially for safety-critical systems. Therefore, we formulate our hypotheses, as follows:

If *a blockchain-enabled framework that allows organizations to trace and manage requirements throughout the software development lifecycle exists,* ***then*** *such a framework can be adopted by organizations to promote cooperation and trust among different parties by means of blockchain technologies.*

To the best of our knowledge, to date, such a framework does not exist. Thus, this will be the first attempt within the novel field of blockchain-oriented RE. Bearing in mind the identified challenges of RT, we aim to design a blockchain-enabled framework to trace requirements throughout the software development lifecycle. By means of this framework, we claim to enhance the accuracy and trust of the RT process and minimize the work of humans, e.g. requirements analysts, in the intrinsically difficult tracing process. Applying the supply chain mindset to the context of SDLC composed of various value creation activities enables us to believe that traceability of requirements may be enhanced by means of blockchain properties and principles, in the following facets:

- A distributed and transparent system to store, access and query RT-related data, independent of the variety of tools used throughout the software development life cycle.
- Guarantee the integrity and immutability of RT data.
- Remove the need for third-party RT tools, enhancing in this way trustworthiness of RT data, which per se are sensitive.
- Shift the responsibility for the management and traceability of requirements from requirements engineer(s), analyst(s) or manager(s) towards each of the stakeholders involved in the software development life cycle.
- Achieve near real-time traceability e.g. a specific developer creates and registers trace links (e.g. requirements to source code), as the developer writes his code.
- Blockchain transactions consisting of trace links will be assessed by validators and if consensus is achieved, e.g. by means of majority voting, they will be part of the append-only blockchain.
- Incentivize stakeholders for creating and validating traceability data.
- Automated compliance to regulations or standards for RT, by means of smart contracts.
- Ensure visibility to customers, who can verify the state of their requirements at any time and detect inconsistencies or defects.

IV. Research Methodology

The following sections will briefly describe the research process, methods and techniques planned to be adopted throughout this PhD project.

A. Research Process

The structure of this research relies on the research process proposed by Oates [39]. This process consists of five stages (See Figure 1) as follows:

- Initiation- define relevant concepts, the problem area, research hypothesis, by means of systematic literature reviews.
- Research paradigm- determine an appropriate research paradigm for the topic of interest.
- Data generation- identify a suite of robust approaches, to generate the data relevant for this research. This consists of identification of reference frameworks in RT, identification of existing challenges in the RT practice and the design of a well-defined framework that solves these challenges. Such a framework will provide a blueprint for blockchain-enabled RE.
- Data analysis and evaluation- identification of appropriate data analysis and evaluation methods. The evaluation of the design consists of two stages. Firstly, the key features of the framework will be validated by expert reviews and

secondly, the proposal will be evaluated in experimental and organizational settings.

- Examination of research hypothesis and questions- research hypothesis will be validated by means of the data generated and evaluated in the previous stages.

B. Research Methods and Techniques

Over two decades, researchers have been proposing solutions that intend to solve the RT problem. Despite these efforts, the problem remains unsolved. To serve to this purpose, this research is expected to contribute to existing knowledge about RT, as well as to have practical implications. We propose a blockchain-based framework as an innovative way to solve the requirements traceability problem. Given that our target is not a specific organization or domain, but all those organizations that are interested in RT, the most appropriate research method is design science research [40]. A triangulation of data generation techniques will be adopted to ensure generalizability of findings, as suggested by Oates [39]. Firstly, the literature with regards to RE, RT and blockchain in SE will be investigated in a rigorous manner, by means of SLR and systematic mapping that rely on the well-grounded guidelines for SLRs in software engineering [41] and guidelines for systematic mappings in software engineering [42].

Secondly, in order to support the development of the framework, mappings among standards and practices will be carried out. Ultimately, the framework will be evaluated in an artificial fashion, by means of experiments and naturalistic fashion [43], by means of qualitative techniques such as interviews, focus groups, documentations and quantitative techniques such as questionnaires. With regards to the qualitative analysis of data, thematic coding analysis will be adopted to extract themes which will be further used in the improvement of the framework.

Figure 1. Research process stages.

V. PROGRESS AND CONCLUSION

This PhD project was approved by the end of November 2019 and is expected to finish in three years. In order to identify the problem domain and the research hypothesis, a general literature review in the fields of RE, RT and blockchain in SE was carried out. In particular, we are writing a paper focused on identifying the factors influencing the use and adoption of automated RT in practice. By doing so, we intend to analyze the underlying problems under the umbrella of RT problem. The identification of these factors may serve as basis to design a holistic solution that aims to address this problem. Based on our preliminary literature review, we believe that blockchain technology may significantly contribute to the design of this solution. Therefore, we are also working on a second paper which consists of a systematic mapping that aims to identify how blockchain contributes to solving problems in software engineering. During the second year, we plan to conduct interviews and surveys with stakeholders and experts in our fields of interest. Afterwards, we will collect and analyze the data from the aforementioned sources and the literature review, in order to design the framework. Once designed, the framework will be subjected to an expert validation. Based on experts' feedback, we will enhance the initial framework design. Finally, during the last year the framework will be implemented in organizational settings and it will be evaluated which will lead to the last improvement cycle.

To wrap it up, we propose a blockchain-enabled framework that intends to trace and manage requirements throughout the software development life cycle. This framework aims to increase the accuracy and trustworthiness of the RT process, by removing the need for third-parties requirements management and/or tracing tools and providing a distributed, yet collaborative environment where a variety of stakeholders agree on standards, tracing strategies, and share artifacts in a confidential manner.

Moreover, this framework will consider other problems, e.g. how to incentivize stakeholders to create and validate trace links in their daily work or how to automate compliance of requirements with standards and strategies by means of smart contracts. We perceive the blockchain-oriented RE as a very promising field, with the potential to disrupt RE in general and finally solve the RT problem in particular.

ACKNOWLEDGMENT

This PhD project is supported by a scholarship from Østfold University College, Halden, Norway. Moreover, I would like to thank Oslo University for accepting this PhD proposal. Finally, I would like to thank Professor Ricardo Colomo-Palacios and Professor Roman Vitenberg for supervising this thesis.

REFERENCES

[1] J. Dick, E. Hull, and K. Jackson, *Requirements engineering*. Springer, 2017.

[2] K. Villela *et al.*, "Towards Ubiquitous RE: A Perspective on Requirements Engineering in the Era of Digital Transformation," in *2018 IEEE 26th International Requirements Engineering Conference (RE)*, Aug. 2018, pp. 205–216, doi: 10.1109/RE.2018.00029.

[3] D. M. Fernández, "Supporting Requirements-Engineering Research That Industry Needs: The NaPiRE Initiative," *IEEE Softw.*, vol. 35, no. 1, pp. 112–116, Jan. 2018, doi: 10.1109/MS.2017.4541045.

[4] B. Nuseibeh and S. Easterbrook, "Requirements engineering: a roadmap," in *Proceedings of the Conference on The Future of Software Engineering*, Limerick, Ireland, May 2000, pp. 35–46, doi: 10.1145/336512.336523.

[5] O. C. Z. Gotel and C. W. Finkelstein, "An analysis of the requirements traceability problem," in *Proceedings of IEEE International Conference on Requirements Engineering*, Apr. 1994, pp. 94–101, doi: 10.1109/ICRE.1994.292398.

[6] S. Murugappan and D. Prabha, "Requirement Traceability for Software Development Lifecycle," *Int. J. Sci. Eng. Res.*, vol. 8, no. 5, pp. 1–11, 2017.

[7] S. Winkler and J. von Pilgrim, "A survey of traceability in requirements engineering and model-driven development," *Softw. Syst. Model.*, vol. 9, no. 4, pp. 529–565, Sep. 2010, doi: 10.1007/s10270-009-0145-0.

[8] S. Maro, J.-P. Steghöfer, and M. Staron, "Software traceability in the automotive domain: Challenges and solutions," *J. Syst. Softw.*, vol. 141, pp. 85–110, Jul. 2018, doi: 10.1016/j.jss.2018.03.060.

[9] B. Wang, R. Peng, Y. Li, H. Lai, and Z. Wang, "Requirements traceability technologies and technology transfer decision support: A systematic review," *J. Syst. Softw.*, vol. 146, pp. 59–79, Dec. 2018, doi: 10.1016/j.jss.2018.09.001.

[10] Y. Udagawa, "An Augmented Vector Space Information Retrieval for Recovering Requirements Traceability," in *2011 IEEE 11th International Conference on Data Mining Workshops*, Dec. 2011, pp. 771–778, doi: 10.1109/ICDMW.2011.27.

[11] J. Shao, W. Wu, and P. Geng, "An Improved Approach to the Recovery of Traceability Links between Requirement Documents and Source Codes Based on Latent Semantic Indexing," in *Computational Science and Its Applications – ICCSA 2013*, Berlin, Heidelberg, 2013, pp. 547–557, doi: 10.1007/978-3-642-39640-3_40.

[12] T. Dietrich, J. Cleland-Huang, and Y. Shin, "Learning effective query transformations for enhanced requirements trace retrieval," in *2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, Nov. 2013, pp. 586–591, doi: 10.1109/ASE.2013.6693117.

[13] C. Ziftci and I. Krueger, "Tracing requirements to tests with high precision and recall," in *2011 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011)*, Nov. 2011, pp. 472–475, doi: 10.1109/ASE.2011.6100102.

[14] S. Pandanaboyana, S. Sridharan, J. Yannelli, and J. H. Hayes, "REquirements TRacing On target (RETRO) enhanced with an automated thesaurus builder: An empirical study," in *2013 7th International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE)*, May 2013, pp. 61–67, doi: 10.1109/TEFSE.2013.6620156.

[15] A. Mahmoud and N. Niu, "Supporting requirements traceability through refactoring," in *2013 21st IEEE International Requirements Engineering Conference (RE)*, Jul. 2013, pp. 32–41, doi: 10.1109/RE.2013.6636703.

[16] W. Wang, A. Gupta, N. Niu, L. Da Xu, J.-R. C. Cheng, and Z. Niu, "Automatically Tracing Dependability Requirements via Term-Based Relevance Feedback," *IEEE Trans. Ind. Inform.*, vol. 14, no. 1, pp. 342–349, Jan. 2018, doi: 10.1109/TII.2016.2637166.

[17] C. Mills, J. Escobar-Avila, and S. Haiduc, "Automatic Traceability Maintenance via Machine Learning Classification," in *2018 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, Sep. 2018, pp. 369–380, doi: 10.1109/ICSME.2018.00045.

[18] D. M. Shahid, S. Ibrahim, and M. Mahrin, "An evaluation of requirements management and traceability tools," *World Acad. Sci. Eng. Technol.*, vol. 78, pp. 596–601, Jun. 2011.

[19] B. Ramesh, "Factors Influencing Requirements Traceability Practice," *Commun ACM*, vol. 41, no. 12, pp. 37–44, Dec. 1998, doi: 10.1145/290133.290147.

[20] F. Blaauboer, K. Sikkel, and M. N. Aydin, "Deciding to Adopt Requirements Traceability in Practice," in *Advanced Information Systems Engineering*, Berlin, Heidelberg, 2007, pp. 294–308, doi: 10.1007/978-3-540-72988-4_21.

[21] A. Kannenberg and H. Saiedian, "Why Software Requirements Traceability Remains a Challenge," *J. Def. Softw. Eng.*, vol. 22, pp. 14–19, Jul. 2009.

[22] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends," in *2017 IEEE International Congress on Big Data (BigData Congress)*, Jun. 2017, pp. 557–564, doi: 10.1109/BigDataCongress.2017.85.

[23] X. Xu *et al.*, "A Taxonomy of Blockchain-Based Systems for Architecture Design," in *2017 IEEE International Conference on Software Architecture (ICSA)*, Apr. 2017, pp. 243–252, doi: 10.1109/ICSA.2017.33.

[24] M. Friedlmaier, A. Tumasjan, and I. M. Welpe, "Disrupting Industries With Blockchain: The Industry, Venture Capital Funding, and Regional Distribution of Blockchain Ventures," Social Science Research Network, Rochester, NY, SSRN Scholarly Paper ID 2854756, Sep. 2017. doi: 10.2139/ssrn.2854756.

[25] S. Porru, A. Pinna, M. Marchesi, and R. Tonelli, "Blockchain-Oriented Software Engineering: Challenges and New Directions," in *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, May 2017, pp. 169–171, doi: 10.1109/ICSE-C.2017.142.

[26] P. Chakraborty, R. Shahriyar, A. Iqbal, and A. Bosu, "Understanding the software development practices of blockchain projects: a survey," in *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, Oulu, Finland, Oct. 2018, pp. 1–10, doi: 10.1145/3239235.3240298.

[27] M. Yilmaz, S. Tasel, E. Tuzun, U. Gulec, R. V. O'Connor, and P. M. Clarke, "Applying Blockchain to Improve the Integrity of the Software Development Process," in *Systems, Software and Services Process Improvement*, Cham, 2019, pp. 260–271, doi: 10.1007/978-3-030-28005-5_20.

[28] M. Beller and J. Hejderup, "Blockchain-based software engineering," in *Proceedings of the 41st International Conference on Software Engineering: New Ideas and Emerging Results*, Montreal, Quebec, Canada, May 2019, pp. 53–56, doi: 10.1109/ICSE-NIER.2019.00022.

[29] M. Król, S. Reñé, O. Ascigil, and I. Psaras, "ChainSoft: Collaborative Software Development using Smart Contracts," in *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems*, Munich, Germany, Jun. 2018, pp. 1–6, doi: 10.1145/3211933.3211934.

[30] V. Lenarduzzi, M. I. Lunesu, M. Marchesi, and R. Tonelli, "Blockchain applications for agile methodologies," in *Proceedings of the 19th International Conference on Agile Software Development: Companion*, Porto, Portugal, May 2018, pp. 1–3, doi: 10.1145/3234152.3234155.

[31] R. P. J. C. Bose, K. K. Phokela, V. Kaulgud, and S. Podder, "BLINKER: A Blockchain-Enabled Framework for Software Provenance," in *2019 26th Asia-Pacific Software Engineering Conference (APSEC)*, Dec. 2019, pp. 1–8, doi: 10.1109/APSEC48747.2019.00010.

[32] O. Gotel *et al.*, "The Grand Challenge of Traceability (v1. 0)," 2011.

[33] N. Niu, W. Wang, and A. Gupta, "Gray Links in the Use of Requirements Traceability," in *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, New York, NY, USA, 2016, pp. 384–395, doi: 10.1145/2950290.2950354.

[34] S. Gayer, A. Herrmann, T. Keuler, M. Riebisch, and P. O. Antonino, "Lightweight Traceability for the Agile Architect," *Computer*, vol. 49, no. 5, pp. 64–71, May 2016, doi: 10.1109/MC.2016.150.

[35] S. Maro, A. Anjorin, R. Wohlrab, and J.-P. Steghöfer, "Traceability maintenance: Factors and guidelines," in *2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE)*, Sep. 2016, pp. 414–425.

[36] R. Wohlrab, E. Knauss, J.-P. Steghöfer, S. Maro, A. Anjorin, and P. Pelliccione, "Collaborative traceability management: a multiple case study from the perspectives of organization, process, and culture," *Requir. Eng.*, Nov. 2018, doi: 10.1007/s00766-018-0306-1.

[37] P. Mäder, P. L. Jones, Y. Zhang, and J. Cleland-Huang, "Strategic Traceability for Safety-Critical Projects," *IEEE Softw.*, vol. 30, no. 3, pp. 58–66, May 2013, doi: 10.1109/MS.2013.60.

[38] P. Rempel, P. Mçder, and T. Kuschke, “An empirical study on project-specific traceability strategies,” in *2013 21st IEEE International Requirements Engineering Conference (RE)*, Jul. 2013, pp. 195–204, doi: 10.1109/RE.2013.6636719.

[39] B. J. Oates, *Researching Information Systems and Computing*. SAGE, 2006.

[40] J. Iivari and J. R. Venable, “Action research and design science research-seemingly similar but decisively dissimilar,” 2009.

[41] B. Kitchenham, “Procedures for Performing Systematic Reviews,” p. 33.

[42] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, “Systematic mapping studies in software engineering,” in *Proceedings of the 12th international conference on Evaluation and Assessment in Software Engineering*, Italy, Jun. 2008, pp. 68–77, Accessed: Apr. 23, 2020. [Online].

[43] J. Venable, “A framework for design science research activities,” in *Emerging Trends and Challenges in Information Technology Management: Proceedings of the 2006 Information Resource Management Association Conference*, 2006, pp. 184–187.