Insights on Transferring Software Engineering Scientific Knowledge to Practice

Talita V. Ribeiro[1], Jeffrey C. Carver[2], Guilherme H. Travassos[1]

[1]Programa de Engenharia de Sistemas e Computação – Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro, Brasil

[2]Department of Computer Science – University of Alabama, Tuscaloosa, USA.

{tvribeiro,ght}@cos.ufrj.br, carver@cs.ua.edu

Abstract. *CONTEXT. In software engineering (SE), aligning research and practice has long been challenging. GOAL. To assist researchers in extracting practical issues from the practical knowledge repositories of SE and making scientific information about SE accessible to practitioners. METHOD. We conducted several empirical studies to determine the causes of the disconnect between research and practice, which makes it challenging for practitioners to seek out and apply scientific knowledge. RESULTS. We obtained data on practitioners' difficulties with finding, comprehending, and evaluating SE scientific knowledge that supported us in creating a set of eight heuristics for conducting practical research in SE.*

Resumo. *CONTEXTO. Na engenharia de software (ES), alinhar pesquisa e prática tem sido um desafio. OBJETIVO. Auxiliar os pesquisadores na extração de questões práticas dos repositórios de conhecimento prático de ES; e em tornar as informações científicas sobre ES acessíveis aos profissionais. MÉTODO. Realizamos vários estudos experimentais para determinar as causas da desconexão entre pesquisa e prática que torna desafiador para os profissionais buscar e aplicar o conhecimento científico. RESULTADOS. Obtivemos dados sobre os desafios dos profissionais para encontrar, entender e avaliar a literatura científica de ES que nos auxiliaram a construir um conjunto de oito heurísticas para a realização de pesquisas práticas em ES.*

1. Introduction

Software engineering (SE) research needs practice to evolve. Since the 1980s (BASILI, SELBY, and HUTCHENS, 1986), empirical research has supported evaluating, predicting, understanding, controlling, and improving software development processes and products. Many software technologies have been developed to support various development process areas, from requirements elicitation to software deployment. One of the most significant things we took away from this period was the realization that the SE research community and the industry must work together to provide high-quality and pertinent study results and proposals (SJøBERG, DYBÅ, and JøRGESEN, 2007).

SE practice can take advantage of scientific production as well. Many academic software technologies have been widely employed in practice since their inception till the present. For instance, Simula, a language for discrete event simulation, is credited as an inspiration for object-oriented programming. Simula was developed in the 1960s as part of a project by scientists at the Norwegian Computing Center (DAHL, 2002). Additionally, academics documented software quality prediction with software metrics in the 1970s (AKIYAMA, 1971) (MCCABE, 1976) (HALSTEAD, 1977), and they are currently included in numerous IDEs and source code analysis tools for various programming languages. Also, software design patterns (BECK and CUNNINGHAM, 1987) (GAMMA *et al.*, 1994) are examples of academic creations that successfully merged with practice.

Despite the reciprocal benefits, SE has struggled to bridge the gap between research and practice from about 1969 to the present (GAROUSI, PETERSEN, and OZKAN, 2016). However, over this time, there have been numerous attempts to facilitate information transfer between these two groups. Researchers were able to grasp the actual issues that arise in software development projects, for instance, by conducting surveys (PFLEEGER and KITCHENHAM, 2001) with practitioners and conducting ethnographies (SHARP, DE SOUZA, and DITTRICH, 2010) in software organizations. Design science (HEVNER and CHATTERJEE, 2010) and action research were introduced into software development projects. These methods assisted practitioners in resolving local problems while using researchers alongside their software teams. Additionally, literature review guidelines (KITCHENHAM and BUDGEN, 2022) were developed and employed in industrial settings to aggregate various study results and offer guidance for software projects. Even so, sharing knowledge using these methods is only used sometimes in the SE industry.

At least in a solo practitioner effort, software companies rarely use scientific findings and results to support decision-making (JEDLITSCHKA, JURISTO, and ROMBACH, 2014). As researchers, we understand that disregarding scientific evidence in practice can result in inadequate technology adoptions that are neither suitable nor applicable to the software project needs (DYBÅ, KITCHENHAM, and JøRGENSEN, 2005). However, claiming that software engineers in the software sector make poor decisions is false. As an illustration, since its debut in 2008, Stack Overflow (ATWOOD and SPOLSKY, 2008) has been a technological forum for professionals to discuss computer programming, exchange tips for developing software, and advance software technologies. Its intense use by software engineers correlates with its popularity, as seen by Alexa's ranking, which ranks the platform among the top 100 most frequented websites globally (KAHLE and GILLIAT, 1996).

We can infer from this situation that the scientific production, as it has been made available to practitioners, has not been sufficient to meet their informational needs. Other research fields have noted this situation (STRAUS, TETROE, and GRAHAM, 2013). Researchers must provide mechanisms for practitioners to apply scientific knowledge more naturally, such as with less restrictive information sources, rather than simply urging them to look for and use scientific productions. In this manner, the use of scientific knowledge as a foundation for decision-making during SE activities can be increased. Based on this scenario, this research work conducted a series of experimental studies to identify why the gap between research and practice still exists even after so much effort from the academic side to bridge it. The findings supported the proposal of research support to produce studies that learn from practice and provide practice with scientific knowledge.

The next sections are organized as follows: Section 2 presents an overview of the context this research is placed, as well as the research goal, questions, and methodology used for their investigation; Section 3 presents the investigation concerning the challenges SE practitioners might encounter while taking scientific knowledge into practice; Section 4 presents the investigation concerning practitioners' information needs; Section 5 presents a set of eight heuristics for conducting practical research on SE; Section 6 presents the final remarks.

2. Research Context, Goal, Questions, and Methodology

2.1. The Context and Goal of this Research

Information diffusion, transfer, and translation are three key ideas that help us comprehend the process of moving information from those who produce it to those who utilize it, from its creation to its application. These concepts are connected and frequently used interchangeably in technical literature; they relate to transmitting information from its creator (the transferor) to its user (the transferee).

While knowledge transfer requires an agreement between the people involved and an active and planned procedure for disseminating and obtaining knowledge, knowledge diffusion refers to spreading knowledge to a target group of users, typically freely and passively (HAMERI, 1996). Additionally, a knowledge transfer is successful once the recipient can apply the knowledge effectively in their surroundings—a requirement not necessary for knowledge diffusion (RAMANATHAN, 2008). Conversely, knowledge translation differentiates from knowledge transfer primarily because of its message characteristics. Budgen, Kitchenham, and Brereton borrowed the knowledge translation concept from medicine to SE. It is defined as "the exchange, synthesis and ethically-sound application of knowledge – within a complex system of interactions between researchers and users – to accelerate the capture of the benefits of research through better quality software and software development processes" (BUDGEN, KITCHENHAM, and BRERETON, 2013). In medicine and SE, the authors advocate for the message to be quality assessed, synthesized, and aggregated before its transfer to reduce bias from individual works. Thus, a solution for knowledge translation in SE should consider all steps among gathering knowledge until its selection for use.

The knowledge exchange flow can happen from research to research, from practice to practice, from practice to research, and from research to practice. In each of these scenarios, there are different types of knowledge and efforts in taking the knowledge from the producer to its user. While analyzing a knowledge flow, one of the most important things to identify is who (producer or user) will take knowledge from one side to the other. The knowledge flow related to taking scientific knowledge from research to practice (Figure 1) and the knowledge flow related to taking practical knowledge from practice to research (Figure 2) are the flows under the focus of this research work. This last flow is important because we must first understand what practitioners need so that the scientific knowledge produced meets their expectations before its translation, transfer, or even diffusion to practice.

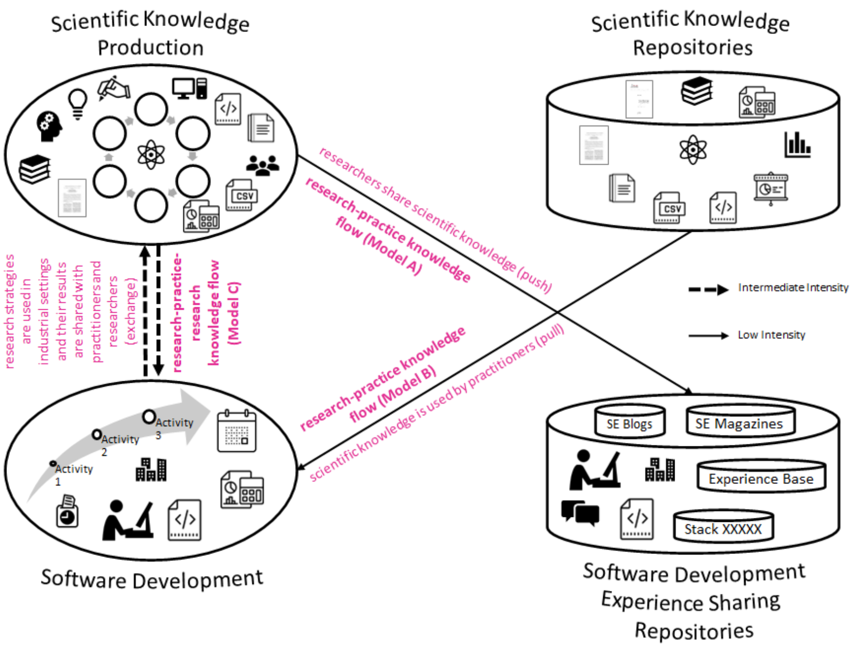

Figure 1. Research-practice knowledge flows – pulling, pushing, and exchange efforts involved

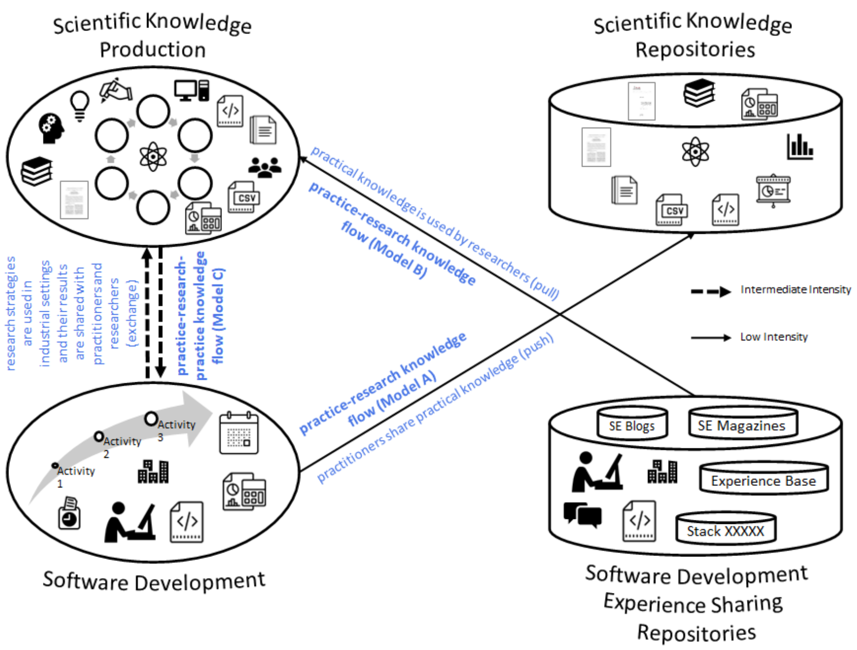

Figure 2. Practice-research knowledge flows – pulling, pushing, and exchange efforts involved

2.2. Research Questions and Methodology

Upon the motivation, problem, and context, a series of studies were carried out to investigate the following primary research question:

PRQ: What to consider while planning, executing, and reporting empirical studies in software engineering to reach practitioners?

To better organize the investigation, the following secondary research questions were stated:

SRQ1: What are the problems faced by those who search for and use SE scientific evidence in practice?

SRQ2: What are the practitioners' information needs that can be used to guide practical research on SE?

As for the methodology chosen to support this research, the scientific knowledge engineering approach (SANTOS and TRAVASSOS, 2016) was selected because it suits the need for knowledge structure required for this type of work. The next sections present the research findings trying to answer the secondary research questions while following the methodology described in (SANTOS and TRAVASSOS, 2016).

The findings will be organized into four assumptions related to the main reasons for intended users do not use a piece of knowledge (BENNETT and JESSANI, 2011):

(i) they do not know the information exists or what action to take in its regard;
(ii) they do not understand the information; what it means, and why it is essential;
(iii) they do not care and see the information as irrelevant, not beneficial to their agenda;
(iv) they do not agree and think the information is misguided or false.

3. Challenges to Taking Scientific Knowledge to Practice

We conducted two studies to see if SE practitioners do not use scientific evidence because <u>*they do not know its existence*</u> and/or because <u>*they do not understand it*</u>. The first study investigates the challenges and pitfalls in gathering SE scientific evidence. The second study is a family of studies examining difficulties in applying SE knowledge to build software technologies. The conclusions of these investigations will provide means to respond to **SRQ1**.

3.1. Challenges and Pitfalls of Surveying Scientific Knowledge in SE

We conducted an exploratory study to learn more about the challenges and pitfalls of surveying scientific knowledge in SE. This study evaluated the design and findings of seven systematic literature reviews (SLR) that addressed the same research issue and were carried out by teams of beginner researchers with similar backgrounds (who have a little more expertise than practitioners in empirical methodology). We pinpointed the major problems that led to the seven SLR plans and reports showing surprising variances. The pitfalls also included pointers for recognizing challenges in SE that can obstruct practitioners' attempts to find scientific information. While details can be found in (RIBEIRO, MASSOLLAR, and TRAVASSOS, 2018), the planning and results of this exploratory investigation are summarized next.

Planning Overview:

Students from the experimental software engineering course at COPPE/UFRJ (2010 and 2012) were called to participate in this exploratory study. They were given lectures on subjects relating to primary and secondary studies in SE and were required to complete several assignments, including developing an SLR research protocol, carrying it out, and reporting the findings. The students (seven D.Sc. and 14 M.Sc.) were organized into seven groups of three, balancing the amount of D.Sc. and M.Sc. among them and assuring the existence of at least one practitioner in each team. The participants were graduate students in their first year, and none had previous knowledge of experimental methods before the course.

The investigation of the SLRs focused on the quality of use cases, having the following research question "Which quality attributes (and measurements used to evaluate such attributes) have been empirically studied for use cases?" The teams were given an initial SLR research protocol with this information and other details related to (i) the search engines to be used; (ii) the inclusion and exclusion criteria; (iii) the initial search terms to support the search string construction; and (iv) ideas for information to be collected from the scientific productions.

The teams should finish the protocol filling within two months, carry out the SLR following their plans, and deliver the findings (quality attributes for use cases). Although neither the research question nor the search engines should be changed, they were advised to modify other parts of the protocol following their understanding of the research topic and question.

The comparison of SLR research techniques and reports was done using Jaccard and Kappa coefficients, and the study covered the usage of search string terms, returned, excluded, and included articles, as well as quality attributes for use cases. Results were anticipated to be similar because the teams underwent the identical initial study protocol and training. Whether or not this was the case, our goal was to pinpoint the likely causes of the discrepancies.

Results Overview:

There is a clear difference between the seven SLRs' plans and findings. For instance, the Black team chose 11 terms, while the Purple team chose 215 terms, and the returned papers ranged from 157 papers in the Pink search to 661 papers in the Black search. The number of quality attributes did not change as much in any case.

We expected more similarities than what we saw. Rarely did the two teams' similarities in the search keywords and returned papers exceed 18% and 12%, respectively. Even though the agreement between SLRs was more related to the articles, a pair of teams agreed to exclude than agreed to include to extract information. Even when all teams in the comparison used the same selection of papers, no two teams' similarities to the reported quality attribute for use cases were 100% identical.

Discussion:

The SLR takes much time to complete. Therefore, two months was not long enough to get better outcomes. The biggest obstacles to SLRs include choosing papers, searching databases, and extracting data (CARVER *et al.*, 2013). While this is true, evidence-based SE is advised to be used in software development environments, and two months can be seen in many cases as a long-time frame for performing a software technology adoption decision in practice.

The differences and difficulties encountered by the teams while conducting SLRs in SE can be attributed to six key causes, they are the lack of (i) experience in the investigated topic; (ii) experience in the research method; (iii) a standard terminology regarding use cases, requirements, and quality attributes; (iv) clearness and completeness of scientific papers; (v) verification procedures to support following an SLR process; (vi) commitment or interest in the research topic.

This study's findings allow us to conclude that even if practitioners could search for scientific knowledge less methodically than students, they would still encounter almost all the same difficulties when doing so. Students had a slight advantage over practitioners in terms of their knowledge of research methodology. It implies that finding the information practitioners seek in research productions is not simple. The situation worsens because most research journals are not free or open access, and the open science movement is still in its infancy (MENDEZ *et al.*, 2020). *It leads to the conclusion that SE practitioners might not know specific scientific knowledge exists and should not try to take scientific knowledge from research to practice.*

3.2. Challenges and Barriers to Understanding and Using Scientific Knowledge in SE

We saw a chance to look more closely at the challenges associated with using evidence to develop a workable solution to a real-world issue, and this investigation allowed us to uncover several barriers and challenges with how scientific knowledge is perceived and applied in SE. While details can be found in (RIBEIRO and TRAVASSOS, 2018) and (RIBEIRO, SANTOS, and TRAVASSOS, Accepted for Publication in 2023), the planning and results of a literature review and a family of studies and their aggregation are summarized next.

Structured Review Overview:

Using the Scopus engine, we conducted a structured literature review, assessed the returning papers, and retrieved the data from the accepted ones. Our goal was to gather data about the impact of source code quality attributes on source code readability and comprehensibility, and that would support the construction of coding guidelines to be used in a software organization. In addition, we intended to identify any data that would help determine the causes of evidence contradictions identified in initial returned works so that we could better assemble the software technology at the end.

The effects of 13 source code attributes on the readability and comprehension of source code were extracted from the accepted papers. Each attribute had at least one measuring procedure being reported in the accepted works, resulting in 94 measurement procedures (either qualitative or quantitative). Five out of 13 presented contradictory evidence, not necessarily concerning the same measurement procedure. We conjectured that two main reasons could explain the encountered differences in effects caused by the same source code attribute: (i) the use of different concepts regarding source code reading and comprehension throughout the primary studies; (ii) the existence of unknown context variables to moderate or mediate the cause-effect relationships.

It took time to locate the reasons for the contradictory findings from the scientific articles and it was even more challenging to combine the results to provide a general conclusion and build the coding guidelines. Most studies utilized the terms "readability" and "comprehensibility" interchangeably, which was evident, in particular, in the many measuring procedures employed to identify these desirable qualities in the source code.

While only a few studies attempted to explain the observed phenomenon using contextual data, most of them hypothesized that programming experience plays a significant role when interpreting the impact of source code attributes on source code readability/comprehensibility.

Family of Studies and their Aggregation Overview:

This fact motivated us to plan, carry out, analyze, and combine three local empirical studies on the effect of the presence of comments, indentation spacing, identifiers length, and code size on source code readability and comprehensibility, taking stratified results according to programming experience (novices and experienced programmers) into consideration. The independent examination of expertise was an effort to pinpoint potential causes for the discrepancies in the technical literature and, concurrently, some pointers on the crucial data that researchers must disclose to support comprehension and application of research findings.

The empirical studies were set up to be conducted in person in the classroom. In the three studies – each with a different group of people from computer courses – the participants received general explanations of source code quality from the perspective of readability and comprehensibility, discriminating against these two concepts. The characterization form asked about software development experience, including code guidelines, writing, review, debugging, correction, and maintenance. For the stratification by novices and experienced, we decided to use the students' self-reported experiences (as recommended by Siegmund et al. (SIEGMUND *et al.*, 2014)) in the different experience dimensions captured in the Likert scale (ranging from 0 – no experience to 5 – vast experience).

The results of all three empirical studies were analyzed independently and in their aggregation form, considering results from novices separately from experienced programmers. The three studies collected quantitative and qualitative data from 66 participants regarding their opinions on reading and understanding Python snippets.

Although the quantitative analysis presents statistical significance related to the comprehensibility of commented source codes and the readability of long identifiers, it does not reveal any information that can contribute to solving the contradictions in the technical literature. Moreover, the data analysis did not support our initial assumption that experience could explain the inconsistencies in the technical literature since the results (especially the aggregated ones) are similar for studies with novices and experienced ones.

Discussion:

Even after conducting a family of studies and combining their findings based on the conjectured theories from the initial works, we could not pinpoint the causes of the contradictory results, making it difficult to comprehend the topic from the combination of the findings. Developing evidence-based software technology to enhance source code quality regarding its readability and comprehensibility is difficult. Many other SE subjects might face this same problem.

The difficulties encountered while trying to use scientific knowledge to build software technology can be attributed to five fundamental causes, they are the lack of (i) standard presentation of the information in the papers; (ii) standard terminology for SE; (iii) theoretical studies in SE; (iv) guidance on reporting measurement procedures and contextual information; (v) a repository for knowledge sharing.

Performing rigorous scientific procedures can be tricky, even for those experienced in research methods. The amount of reasoning required to extract, interpret, and synthesize useful scientific knowledge from scientific productions in a problem-solving situation is significant, which should not happen if scientific productions were intended to provide information to practitioners. *It leads to the conclusion that SE practitioners might not understand scientific reports and that scientific articles might not be the best way to present information to them.*

4. Understanding the Information Needs of SE Practitioners

To identify if SE practitioners do not use scientific evidence because *they see the information as irrelevant* and/or *disagree with it*, we analyzed works on the relevance and credibility of SE knowledge to practice. In addition, we carried out two investigations on SE practical questions and answers from an important Q&A forum of SE intending to identify information that seems relevant and credible to SE practitioners. The conclusions of these investigations will provide means to respond to **SRQ2**.

4.1. Scientific Knowledge Relevance and Credibility to Practice

Several works have investigated the relevance of SE scientific knowledge to practitioners. Practitioners believe research produces interesting ideas/proposals, but scientific knowledge is not among their main needs. The fact that they measure a scientific production's relevance (or even credibility) based on whether a practitioner participated in it says a lot about their perception of research produced in labs. The negative comments on research works are usually related to the lack of necessity of it to practice, the lack of fitness to practice, the difficulty in applying it in practice, the real impact of it to practice, and the differences in the context of its application to what exists in practice. In summary, the problem relevance, the solution utility, the solution impact, and the source of the data/results (whether from practitioners or not) are important to practitioners when assessing scientific knowledge as relevant and credible.

To understand what is considered relevant and credible to practitioners in their daily activities, we conducted two investigations on Stack Exchange questions and answers, as we summarize next. Our main goal with these studies was to identify information important to guide researchers while performing and reporting empirical studies in SE.

4.2. Thematic Analysis of Questions Pattern from Stack Exchange

Stack Exchange is a platform offering a collection of 176 forums for Q&A about various topics. Its main forum, Stack Overflow, has more than 21,286,479 questions and 31,692,495 answers on several topics related to software development and maintenance, especially programming. Certainly, it is unfeasible to manually analyze all the questions and their answers from any Stack Exchange forum. Therefore, we had to plan a strategy for narrowing the number of questions/answers that should be analyzed without hampering the study's main goal.

The goal of the thematic analysis of questions from Stack Exchange was to identify relevant SE topics and related questions according to practitioners. To narrow the number of questions to analyze, we used a set of strategies to appraise relevance and sample the questions. From our point of view, a question is relevant whenever (i) it matches the forum's expectations; (ii) it is about one of the most important topics of the forum; (iii) practitioners like it, and (iv) practitioners try to answer it. We understand that there are other perspectives of relevance, and a single question might represent an important issue to be addressed by research. However, this strategy was used to narrow down the number of questions to analyze. After identifying relevant questions from specific important forums, we sampled them, leading to 380 questions to perform a thematic analysis.

As a summary of the results from this analysis, the concepts we identified from the relevant questions are mostly related to the phases of design and coding of the software life cycle: programming languages, software architecture and design, programming practices, databases, software testing, object-oriented development, and web software development. Also, while analyzing the questions, we created meta-questions based on initial research questions formulated for each original question. It helped us understand the main structure of practical questions and their relation to research questions on the topic. We came across 25 meta-questions. We identified that some questions were expecting answers that would describe an intervention. Others were expecting some comparison between the two interventions. We concluded that each meta-question could be mapped to research purposes presented in works like (BASILI, SELBY, and HUTCHENS, 1986) and (WOHLIN *et al.*, 2012). Characterization, comparison, evaluation, and understanding were the purposes we identified from the final list of meta-questions. We identified 20 codes related to interventions/comparisons in forums' questions related to development and methodology (framework, method, paradigm, practice, tool, among others) and product (algorithm, programming language, and others). As for the common mentioned expected outcomes, all product quality characteristics from (ISO/IEC, 2011) appeared in the questions. Most practitioners seek support in achieving functional suitability and performance efficiency. Regarding quality in use, the questions mention effectiveness, efficiency, and satisfaction. The analysis raised one new outcome not related to ISO: competitiveness in terms of cost/price and throughput/velocity.

4.3. Thematic Analysis of Answers Pattern from Stack Exchange

The thematic analysis of answers from Stack Exchange aimed to identify contextual information practitioners uses while answering practical questions and the common way to communicate practical knowledge in these forums. We decided to analyze relevant answers from Stack Exchange, like what was performed with the questions. Apart from the information, we analyzed the types of arguments used in the answers that might make the moderation of the forums accept them, and the community score them highly. A total of 141 answers were analyzed.

As a summary of the results from this analysis, we identified a list of contextual information presented in the answers related to the organization (size of the team, throughput, velocity, and others), product (system type, system complexity, and others), stakeholder (experience), development and methodology (development practices, and others), and business and market (cost/price). Concerning the arguments, we noticed that the arguments that the answers creators provide are usually based on personal experience/opinions. Thus, the argumentation schemes presented along with the answers are basically: an argument based on cases, an argument from analogy, an argument from alternative, an argument from example, an argument from expert opinion, an argument from popular opinion, a statement from widespread practice, and practical reasoning.

As for the style of the answers, all are presented in written format, and very few provide images or drawings along with them. However, many users provide external information (links) that complement their answers. Also, the answers are usually organized in sections and/or in items/numbers, and the creators of the answers take advantage of format styling (e.g., bold, italic, underline, and font size) to emphasize important information.

The primary distinction between practitioners' and academics' responses is the candor of the presented data. There is a concern with giving the complete technique used in research presentations to make a study repeatable and fully disclose the findings. However, practitioners are not required to explain how they came to possess the knowledge they deliver. Additionally, practitioners give far more direct counsel, possibly because there is little concern over making forceful claims.

Although the answers are specific, they may or may not relate to the question's stated context. Highly scored responses typically discuss how a specific intervention is used in several circumstances, describing the various outcomes and side effects that can occur in each. The practitioners' own real-world experiences typically support the statements. Undoubtedly, the community attests to its expertise in the subject under debate, typically demonstrated by the examples given in the answers.

Although we are not expected to come up with the same solutions as practitioners, we may learn from them, particularly in how they convey their findings. Additionally, conducting context-driven studies focusing on somewhat varied settings is an intriguing method for offering comprehensive and all-encompassing answers. It challenges the idea that all relevant contextual information must be reported for a study to be complete. Rather, many different settings should be assembled to study a single intervention.

5. Support for Researching Real SE Issues

We determined the value of SE topics and information to practitioners through two studies in the Stack Exchange community. Additionally, we could pinpoint the crucial data, categorization of the arguments, and modes of presentation used by practitioners in acceptable answers to practical questions. Our core premise is that for knowledge dissemination, transfer, and even translation to practice to be successful, researchers must incorporate into their research efforts a means of obtaining practical issues from practice, researching them, and relaying the findings to practitioners' channels of communication.

A total of 8 heuristics were proposed to guide researchers in researching practical issues and reporting scientific results using repositories of SE practical knowledge. The heuristics were created based on the studies' execution and conclusions throughout this doctoral research. The heuristics are related to problem identification, research question formulation, study design, and publication of the results. In some sense, along with the studies' results, they are meant to answer the main research question of this work (PRQ). An overview of them can be seen in Figure 3.

Figure 3. The eight practical research heuristics overview

Some heuristics can be time-consuming and error-prone to apply when using Q&A forums. After conducting two observational studies with little computing support to assess the heuristics, we concluded that a computational infrastructure would be beneficial to use Q&A forums as a source of real-world knowledge to support research works.

The heuristics formulated was the primary source of requirements for building a computational infrastructure. In addition, the tool considers part of the data dumps provided continuously by Stack Exchange and offers specific functionalities for acquiring practical knowledge that can support research. Once we had the availability of the data and a tool to support handling it, we decided to add natural language processing to support the application of some of the heuristics presented.

6. Final Remarks

The expectation that practitioners can invest effort in software development to search and use scientific knowledge from its sources is unrealistic. They are not trained in scientific methods and are not used to following the rigorous process required in research (JURISTO and MORENO, 2001). In addition, practitioners and researchers see software development problems and solutions differently (GAROUSI, PETERSEN, and OZKAN, 2016).

Gathering everyday SE concerns from practice and putting effort into getting scientific productions to practitioners in a form they can easily access, understand, and evaluate are better ways to achieve some success in having scientific knowledge employed in practice. The key to success in this situation is knowing what to transfer, where to deliver it, and how to present it.

This doctorate research presented several insights on transferring SE scientific knowledge to practice that can be used to guide future works intending to overcome difficulties in collaborations between research and practice or even to improve the planning, execution, and reporting of empirical studies in SE. These insights led us to formulate heuristics to support identifying researchable topics and questions from SE Q&A forums and transforming practical questions into research questions, research purposes, and search strings to guide context-driven research on practical issues that can be reported directly to practitioners. More details on the work can be found in (RIBEIRO, 2022).

7. Acknowledgements

The authors thank CAPES and CNPq for supporting this research. Prof. Travassos is a CNPq Researcher and CNE FAPERJ.

8. References

AKIYAMA, F. **An example of software system debugging**. Proceedings of IFIP Congress. Ljubljana: North-Holland. 1971. p. 353-359.

ATWOOD, J.; SPOLSKY, J. **Stack Overflow**, 2008. Available at: <https://stackoverflow.com/>. Access Date: January 2022.

BASILI, V. R.; SELBY, R. W.; HUTCHENS, D. H. Experimentation in software engineering. **Transactions on Software Engineering**, New York, SE-12, n. 7, July 1986. 733-743. DOI: 10.1109/ICSE.1996.493439.

BECK, K.; CUNNINGHAM, W. **Using pattern languages for object-oriented programs**. Proceedings of the Workshop on Specification and Design for Object-Oriented Programming. Florida: ACM. 1987.

BENNETT, G.; JESSANI, N. **The knowledge translation toolkit - bridging the know-do gap:** a resource for researchers. New Delhi: Sage, 2011. 312 p.

BUDGEN, D.; KITCHENHAM, B.; BRERETON, P. **The case for knowledge translation**. Proceedings of the International Symposium on Empirical Software

Engineering and Measurement. Baltimore: IEEE. 2013. p. 263-266. DOI: 10.1109/ESEM.2013.41.

CARVER, J. C. et al. **Identifying barriers to the systematic literature**. Proceedings of the International Symposium on Empirical Software Engineering and Measurement. Baltimore: IEEE. 2013. p. 203-213. DOI: 10.1109/ESEM.2013.28.

DAHL, O.-J. The roots of object-oriented programming: the Simula language. **Software Pioneers**, Berlin, 2002. 78-90. DOI: 10.1007/978-3-642-59412-0_6.

DYBÅ, T.; KITCHENHAM, B.; JøRGENSEN, M. Evidence-based software engineering for practitioners. **IEEE Software**, 22, n. 1, 10 January 2005. 58-65.

GAMMA, E. et al. **Design patterns:** elements of reusable object-oriented software. 1st. ed. California: Addison-Wesley, 1994. 416 p.

GAROUSI, V.; PETERSEN, K.; OZKAN, B. Challenges and best practices in industry-academia collaborations in software engineering: a systematic literature review. **Information and Software Technology**, Amsterdam, 79, November 2016. 106-127. DOI: 10.1016/j.infsof.2016.07.006.

HALSTEAD, M. H. **Elements of Software Science**. New York: North Holland, 1977.

HAMERI, A.-P. Technology-transfer between basic research and industry. **Technovation**, Amsterdam, 16, n. 2, February 1996. 51-57, 91-92. DOI: 10.1016/0166-4972(95)00030-5.

HEVNER, A.; CHATTERJEE, S. **Design research in information systems:** theory and practice. New York: Springer, 2010. DOI: 10.1007/978-1-4419-5653-8.

ISO/IEC. **ISO/IEC 25010 - Software engineering - Software product quality requirements and evaluation (SQuaRE) - system and software quality models**. Geneva, p. 34. 2011.

JEDLITSCHKA, A.; JURISTO, N.; ROMBACH, D. Reporting experiments to satisfy professionals' information needs. **Empirical Software Engineering**, New York, 19, n. 6, December 2014. 1921–1955. DOI: 10.1007/s10664-013-9268-6.

JURISTO, N.; MORENO, A. M. **Basics of software engineering experimentation**. 1. ed. New York: Springer, 2001. 396 p. DOI: 10.1007/978-1-4757-3304-4.

KAHLE, B.; GILLIAT, B. **Alexa Intenet**, 1996. Available at: <https://www.alexa.com/siteinfo/stackoverflow.com>. Access Date: January 2022.

MCCABE, T. J. A complexity measure. **Transactions on Software Engineering**, New York, SE-2, n. 4, December 1976. 308–320. DOI: 10.1109/tse.1976.233837.

MENDEZ, D. et al. Open science in software engineering. In: FELDERER, M.; TRAVASSOS, G. H. **Contemporary empirical methods in software engineering**. Cham: Springer, 2020. p. 477-501. DOI: 10.1007/978-3-030-32489-6.

PFLEEGER, S. L.; KITCHENHAM, B. A. Principles of survey research - Part 1: turning lemons into lemonade. **ACM SIGSOFT Software Engineering Notes**, New York, 26, n. 6, November 2001. 16-18. DOI: 10.1145/505532.505535.

RAMANATHAN, K. **An overview of technology transfer and technology transfer models**. Asian and Pacific Centre for Transfer of Technology. Paris, p. 28. 2008.

RIBEIRO, T. **Insights on Transferring Software Engineering Scientific Knowledge to Practice**. DSc Thesis: COPPE / Universidade Federal do Rio de Janeiro, 2022. 179 p.

RIBEIRO, T. V.; MASSOLLAR, J.; TRAVASSOS, G. H. Challenges and pitfalls on surveying evidence in the software engineering technical literature: An exploratory study with novices. **Empirical Software Engineering**, New York, 23, June 2018. 1594–1663. Available at: <http://rdcu.be/xNku>. DOI: 10.1007/s10664-017-9556-7.

RIBEIRO, T. V.; SANTOS, P. S. M. D.; TRAVASSOS, G. H. On the investigation of empirical contradictions - aggregated results of local studies on readability and comprehensibility of source code. **Empirical Software Engineering**, New York, Accepted for Publication in 2023.

RIBEIRO, T. V.; TRAVASSOS, G. H. Attributes influencing the reading and comprehension of source code – discussing contradictory evidence. **CLEI Electronic Journal**, 21, n. 1, April 2018. 1-33. DOI: 10.19153/cleiej.21.1.4.

SANTOS, P. S. M. D.; TRAVASSOS, G. H. Scientific knowledge engineering: a conceptual delineation and overview of the state of the art. **The Knowledge Engineering Review**, Cambridge, 31, n. 2, March 2016. 167-199. DOI: 10.1017/S0269888916000011.

SHARP, H.; DE SOUZA, C.; DITTRICH, Y. **Using ethnographic methods in software engineering research**. Proceedings of the International Conference on Software Engineering. Cape Town: IEEE. 2010. p. 491-492. DOI: 10.1145/1810295.1810445.

SIEGMUND, J. et al. Measuring and modeling programming experience. **Empirical Software Engineering**, New York, 19, n. 5, December 2014. 1299-1334. DOI: 10.1007/s10664-013-9286-4.

SJøBERG, D. I. K.; DYBÅ, T.; JøRGESEN, M. **The future of empirical methods in software engineering research**. Proceedings of the Future of Software Engineering Symposium. Minneapolis: IEEE. 2007. p. 358-378. DOI: 10.1109/fose.2007.30.

STRAUS, S. E.; TETROE, J.; GRAHAM, I. D. **Knowledge translation in health care:** moving from evidence to practice. Hoboken: Wiley, 2013. DOI: 10.1002/9781118413555.

WOHLIN, C. et al. **Experimentation in software engineering**. Heidelberg: Springer Berlin, 2012. 236 p. DOI: 10.1007/978-3-642-29044-2.