Methodology Matters: Mapping Software Engineering Research Through a Sociotechnical Lens

by

Courtney Bornholdt
B.Eng., Royal Military College of Canada, 2016

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Methodology Matters:
Mapping Software Engineering Research Through a Sociotechnical Lens

by

Courtney Bornholdt
B.Eng., Royal Military College of Canada, 2016

Supervisory Committee

Dr. Margaret-Anne D. Storey, Supervisor
(Department of Computer Science)

Dr. Neil Ernst, Departmental Member
(Department of Computer Science)

ABSTRACT

As software engineering is a socio-technical research field, there is a myriad of research strategies and data sources that researchers need to consider when designing their studies. These choices determine different tradeoffs in terms of generalizability, realism, and control, among other aspects of research quality. It is not possible to create a perfect study, so these strengths and weaknesses are acceptable at the study level; however, when a research community's collective body of work suffers from an imbalance in these tradeoffs it can negatively impact overall research quality.

Through this thesis, I investigate the research strategies and data sources that are used by the software engineering research community, and reflect on how this may affect aspects of research quality in our collective body of work. I apply Runkel and McGrath's models of research strategies and data sources to the software engineering domain through a systematic mapping study of three years of International Conference on Software Engineering (ICSE) proceedings and a mixed-methods survey of the authors of these papers.

I found that a majority of papers report computational studies relying on trace measures rather than active human participation, showing an imbalance where generalizability and realism are prioritized over control. Through my survey, I confirmed that researcher participants explicitly prioritized realism and generalizability over control, impacting their research design choices. This imbalance in prioritization has the potential to lead to a collective failure to control for extraneous factors in the measurement of human behavior in software development, and without understanding what causes the behaviors we measure, we cannot fully understand why certain approaches and techniques work better than others, thus slowing our ability to advance as a research domain. Therefore, I present a call to action for the community to critically examine and discuss the issues raised by this research, and implement changes to increase the quality and diversity of our future work as a community.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

There are a number of wonderful people that I would like to thank:

My husband, Jordan, for so many reasons. My family, for always being there and supporting me, and my mom in particular for helping me edit this thesis. My friends, for listening to me rant and ramble when I wanted to do anything but transcribe another interview or code another text file. Carly, for being my sounding board, my sushi date, my thrifting partner, my office-mate, and my friend. Jorin, for all of our heart-to-heart talks and walks around campus that got me through writing this thesis, and teaching me to love plants. Alexey, for always pushing the envelope and never letting me settle for work that was anything less than exceptional. Eirini, for all of your help with my research, and giving me an excuse to pursue my love for baking. Peggy, for all of your guidance and patience through the last two years. You helped me fall in love with my own work again and again, and I hope I made you proud. Cassie, for keeping us all from exploding every day. The CHISEL group, for eating my baked goods and listening to my rants about the importance of humans in research. Neil Ernst and Janet Siegmund, for all of your hard work as part of my examining committee. Greg Phillips, for all of the hard work you did to get me here in the first place. The Natural Sciences and Engineering Research Council, for graciously funding me with a scholarship. My survey participants, for taking the time to fill out a survey that was probably a little too long and complex, and providing me with wonderful insights. And finally, my kitties, Archie and Veronica, for always being down for a cuddle.

DEDICATION

Dedicated to the love of my life, my best friend, my husband, Jordan Bornholdt.

Chapter 1

Introduction

Software engineering research centers around understanding how to improve software development, where software developers and other technical and non-technical stakeholders interact with each other and with technological systems to create and maintain software. This means that software engineering is a *sociotechnical* field, where researchers have to account for both social and technical aspects of software development in their work. Understanding complex sociotechnical contexts requires the use of a wide variety of research methods and positions software engineering in the interdisciplinary fields of engineering, computer science, mathematics, sociology, and psychology.

As a result, software engineering inherits different philosophical views, approaches, and research methods from a variety of fields: (1) quantification and the measurement of systems are inherited from engineering; (2) algorithms, theories, and proofs are inherited from computer science and mathematics; and, (3) empirical methods of inquiry, inference, and interpretation are inherited from sociology and psychology. This raises a timely and important introspective question: *How does software engineering research, at a community level, use methods that capture the social aspects of the sociotechnical endeavor that is software development?*

The best way to select appropriate research methods for a study is a recurring topic of debate, with researchers publishing conflicting works about the benefits and drawbacks of various choices in a research design. Books and articles that aim to provide guidance to others discuss various research methods, methodologies, strategies, and data collection methods without a common taxonomy, and often use terms at different levels of abstraction. This further complicates our ability to communicate our work effectively in our publications as well as evaluate the rigor of studies when acting

as reviewers. Despite the confusion that may be caused by a lack of common terminology, it is an important issue to address, and the choice of methodology matters because it greatly impacts a study's advantages and its limitations. McGrath argued that to *"understand empirical evidence, its meaning, and its limitations, requires that you understand the concepts and techniques on which that evidence is based."* [27]

Earlier work [39, 42, 51, 32] has taken an introspective look into how SE research is conducted to provide guidelines, calls to action, and promote healthy, collective reflection within the research community. In the same vein, we aim to understand and provide an outlook on how the software engineering research community currently approaches studying the *social aspects* of software engineering. We emphasize that we do not aim to make any claims about the goals of *individual researchers* and how the methods used serve those goals, but rather the implications of our method choices at a community level on our ability to address and understand the social aspects of software development.

To reach this goal, we conducted a meta-study examining the *research strategies* and *data sources* reported in studies published within the International Conference on Software Engineering (ICSE) community. We chose to use Runkel and McGrath's models [33] of research strategies and data sources, originally developed to guide research on human behavior in psychology and sociology, as a lens to understand how research produced by the software engineering research community captures the *social perspective* of software development. McGrath [27] saw research methods as *"bounded opportunities"*—choosing a specific method provides opportunities not available with other methods, but also introduces inherent limitations. Their model emphasized that research strategy choice involves trade-offs in *generalizability*, *realism*, and *control*. We are also inspired by how they classify data sources according to human participant involvement in the research process, as this classification lens identifies further advantages and disadvantages that come from the involvement of humans (or lack thereof) in our research, regardless of research strategy.

1.1 Motivation

I was motivated to conduct this research out of a desire to inspire a change in the research community. Some topics in software engineering are sociotechnical endeavors, so it is important when researching these topics to consider both social and technical aspects of software development before making claims about generalizability, realism,

or control. As I began to read the research papers published in prominent software engineering venues, I began to observe that the overwhelming majority of papers had the same general structure. They started by indicating a problem faced by developers, presenting a tool or technique they created to solve the problem, and describing how they evaluated the performance of the tool by applying it to a software artifact, like a series of repositories or a dataset of some type of bug. These papers left me asking the same question: *did the tool actually solve the problem?* Without implementing and evaluating the tool in a real development context, it was unclear whether the tool actually helped the developers whose problems motivated the researchers. Only a handful of papers involved active human participation in the creation or evaluation of their tools, and those papers seemed so much more impactful to me because they showed the benefits and limitations of the tools when used by real developers.

My observations left me with a number of questions that I wanted to answer: How often do we actively engage humans in software engineering research? Does including humans in software engineering research improve some aspect of research quality? If my observations reflect reality, that we do not often include humans in our research despite the potential to positively impact the quality of our work, why does this occur? Can anything be done to solve these problems to improve the quality of our collective body of work?

These questions prompted me to investigate the software engineering research community, the studies we publish, and why we make these choices in order to identify issues and present a call to action. My work is motivated by the desire to give the community the tools and information necessary to start a conversation about human participation in software engineering research and change how we think about research quality.

1.2 Research Questions

To investigate these issues, I apply Runkel and McGrath's models to the International Conference on Software Engineering (ICSE) as a sub-community of the overall software engineering research domain. I use two empirical studies, a systematic mapping study and a survey, to answer the following research questions:

RQ1: What research strategies and data sources are described in research published at ICSE?

RQ2: Why do authors choose the research strategies and data sources they describe in their papers?

RQ3: What is the "balance" between generalizability, control, and realism in research published at ICSE?

1.3 Contributions

This thesis makes three major contributions to the software engineering research community.

Adaptations of Runkel and McGrath's models for the software engineering domain.

In this thesis, I describe how I adapt Runkel and McGrath's models of research strategies and data sources to the software engineering domain. This provides a taxonomy for discussing and describing research at the strategy level and a way of separating our sources of data based on the level of human involvement in their creation. These models can be used to design a series of studies that use complementary research strategies and data sources to triangulate findings more effectively and maximize generalizability, realism, and control. The models can also be used similarly to our work by applying them as a lens to understand the collective research output of a sociotechnical research domain or subcommunity within the software engineering domain.

A snapshot of research strategy and data source choices in the software engineering research community.

In the thesis, I describe two empirical studies, a systematic mapping study and a survey, whose combined findings show the current state of the software engineering research community from a sociotechnical perspective. By classifying ICSE papers based on our adaptations of Runkel and McGrath's models I show the distribution of various research strategies and data source types and show how this may impact the levels of generalizability, realism, and control in our collective body of work. I also suggest possible reasons for this distribution through qualitative analysis of survey responses, where authors elaborated on why they chose the research strategies and data sources reported in their papers.

A reflection on our perceptions and biases towards generalizability, realism, and control in software engineering.

Finally, I reflect on a number of issues raised by the empirical studies. I discuss a number of potentially problematic situations that we discovered through our analysis, and how I believe they may affect the choices we make in our research and the levels of generalizability, realism, and control in our collective body of work. I then call for the community to reflect on these issues, our shared interests and goals for the future of software engineering, and how we must proceed to create increasingly rigorous and impactful research that addresses these issues.

1.4 Thesis Outline

This thesis is divided into a number of chapters. In Chapter 2, I explain my background as a researcher to give context to the decisions I made throughout the research process and the different ways in which my views may have influenced my findings. I outline my overarching research process, as this thesis discusses one of two separate investigations that were conducted concurrently that may have influenced each other. I also discuss how research efforts were divided between collaborators, as this research was a team effort.

In Chapter 3, I explain the origins and history of software engineering research, which provides context for understanding the research methods used by the software engineering community. Then, I discuss the related work that has both informed and motivated the research outlined in this thesis, as well as the seminal works that form our state of the art in understanding how to conduct rigorous studies in software engineering. Finally, I briefly introduce the source of the research lens that forms the theoretical underpinning of this work, as well as one previous application of this lens to software engineering and the limitations of this work.

In Chapter 4 I explain the research lens we developed as part of our research contribution and for use in the systematic mapping study and survey. This is followed by Chapter 5 where I describe the methodology for this research. This chapter is broken down into a number of subsections: a systematic mapping study, survey design, survey dissemination and data collection, combined analysis, and a member checking phase.

In Chapter 6 I outline the findings of the studies, and discuss possible explanations for the results that are grounded in the data. We show the relevance of these findings

in Chapter 7 where I discuss the implications of this work, including recommendations and open questions for discussion for the software engineering research community.

I address the limitations of this research in Chapter 8. Finally, I summarize the research as a whole, identify areas for future work, and conclude the thesis in Chapter 9.

Chapter 2

My Research Background and Worldview

A researcher's background influences their perceptions and decision-making processes, and thus it can potentially create biases within their work. In this section, I detail my epistemological stance, experiences, and worldview, which helped to shape my perceptions prior to the start of my research. As this research was conducted in tandem with a related study, I also describe how the other study motivated and may have influenced this research. Finally, as this research was conducted as part of a team, I discuss the individual roles of each team member and how the team dynamic limited the potential for researcher bias.

2.1 My Research Stance

In order to understand my research decisions and actions described in this thesis, it is first important to understand my worldview and experiences as a researcher. This includes my opinions regarding what constitutes knowledge and how knowledge is constructed, and how my life experiences have shaped me as a person and affected my perceptions of others and of the topic that I am studying.

2.1.1 My Experience and Worldview

My undergraduate degree is in computer engineering, so I had little exposure to social research or philosophical discussions about knowledge. I was firmly postpositivist until I started research at the University of Victoria. Post-positivism is the belief

that "knowledge is conjectural (and anti-foundational) - absolute truth can never be found. Thus, evidence established in research is always imperfect and fallible" and "causes probably determine effects or outcomes" [11]. Post-positivists tend to study humans by "[reducing ideas] into a small, discrete set of ideas to test, such as the variables that constitute hypotheses and research questions", and heavily rely on the use of numbers and statistical analyses [11]. Post-positivism forms the traditional epistemological underpinning of modern physical sciences and engineering.

At that point, having interest in human-computer interaction research topics, I took a course on qualitative research taught by one of our colleagues in the Educational Psychology and Leadership Department. I was introduced to the concepts of epistemology, ontology, and axiology, and how philosophical underpinnings can affect research. The course exposed me to a number of examples of studies from sociology, psychology, social work, education, and other disciplines. I saw the valuable insights that these researchers were able to produce by investigating social issues in a very holistic and constructivist manner, and I gained an appreciation for this type of work in human behavioral research.

Social constructivism is the belief that "individuals seek understanding of the world in which they live" because knowledge and truth are socially constructed. For studies conducted this way, "the goal of research [...] is to rely as much as possible on the participants' views of the situation being studied" in order to "look for the complexity of views rather than [narrow] meanings into a few categories or ideas" [11]. Constructivism is common in social science disciplines such as sociology. As I began to appreciate the benefits of a constructivist approach to social science research, my views on knowledge began to shift from post-positivistic to pragmatic. Pragmatism is the belief that "knowledge claims arise out of actions, situations, and consequences rather than antecedent conditions" [11], where the nature of the problem dictates the research approach that should be used. As I was originally educated from a post-positivist perspective, I can identify with researchers who conduct purely positivistic work and I understand how they might perceive more constructivist research.

2.1.2 My Position on Knowledge

My position on knowledge most closely resembles pragmatism. Pragmatism has varying definitions depending on the source, so I will discuss pragmatism according to descriptions of different views on knowledge from Creswell's work[10]. He describes a

number of knowledge claims associated with four overarching theoretical standpoints, including postpositivism, constructivism, advocacy research, and pragmatism. The part of pragmatism that I particularly identify with is the first claim made by Creswell, that "Pragmatism is not committed to any one system of philosophy and reality"[10]. In general, pragmatists are concerned with problems and select the research methods that are the most appropriate for their research questions, including mixed method approaches. I identify with this assertion about pragmatists, but with social research I am constructivist-leaning. This means that I am more likely to select a holistic, qualitative approach that considers contextual factors to answer a question about human behaviour rather than statistical approaches. This is important because my research questions surround a complex social situation, and this influenced how I seek a deep understanding of the social and historical context surrounding social research in the software engineering research community as part of my research. As a consequence of my stance, I do not highly value statistical information as part of a qualitative study. I believe that it may often be misleading, as it strips away contextual factors, so I intentionally refrain from including numerical data when I describe qualitative findings in this thesis.

2.2 Research Process

As part of my degree, I completed two separate studies that were conducted in tandem to investigate the same topic, so it is important to discuss how progress within each study affected decisions and perceptions within the other. While my interview study is not discussed as a part of this thesis because it is based on different goals, elements of its progression affected and helped motivate this work.

This body of work was originally motivated by an investigation into how published work in software engineering refers to qualitative research methods, which suggested that qualitative research methods were useful in software engineering because they were able to explore complex social phenomenon more effectively than quantitative methods. This investigation also showed that there were various challenges associated with doing qualitative work that resulted in very few researchers utilizing these methods. I used primarily older research papers to inform my findings, which limited my understanding of the problem because it was unclear whether these issues persisted. Therefore, I designed an interview study to investigate the current experience of a qualitative researcher in software engineering to see whether or not these barriers ex-

ist, following an action research for societal change methodology aimed at identifying barriers to producing qualitative research and proposing solutions for the software engineering research community to make qualitative research more prominent.

The initial phases of this interview study indicated that when software engineering researchers want to publish their qualitative papers they tend to target ICSE. This, combined with an interest in updating older statistics about the publication rates of qualitative papers [39], led me to begin investigating the publication rates of qualitative papers at ICSE. The interview study also indicated that my participants conduct much more than just qualitative work and that they also conduct a wide variety of quantitative and mixed methods studies in their work that tended to involve humans. This motivated me to also investigate the level of human involvement in the studies published at ICSE.

My first step in investigating ICSE proceedings was to conduct a preliminary classification of two years of proceedings, categorizing research papers by type of methodology (qualitative, mixed methods, or quantitative) and the level of human involvement (no human involvement, humans as informants, humans as evaluators). The findings from this analysis showed a low amount of qualitative or mixed methods papers, that the majority of papers did not involve human participants, and that this involvement was usually as a tool for evaluation of an approach rather than to learn about some aspect of human perception or behaviour. At this point, we felt that this analysis was not telling the full story, and so we sought out another way to understand this problem. My advisor, Dr. Storey, suggested analyzing ICSE proceedings through a sociotechnical lens using McGrath's 1995 paper on various social science research methods and strategies[27]. After this point, while we continued to conduct analysis on the interviews, the work with ICSE proceedings became the primary focus of my research. During the design, data collection, and analysis phases of this work I was still working on the interview study, and my ideas and findings from the interviews may have influenced my perceptions in this work through researcher bias. We mitigated this potential researcher bias through the use of a highly collaborative research team environment, discussed next.

2.3 The Research Team

This work was conducted as a team effort, and this team dynamic affected the choices that were made during the research process. The research team was made up of four

members, and each played a different role in the research process. These members, and their roles, were:

- **Courtney Bornholdt** - Principal researcher, master's student. I conducted the majority of the research tasks throughout all phases of the project, from preliminary design to reporting. For all research decisions, I proposed my ideas to the rest of the research team for discussion before acting on them and changed plans where necessary to reflect the judgment of the team as a whole.

- **Alexey Zagalsky** - PhD student. He acted as the main consulting researcher throughout all phases of the project. He did not conduct many research tasks himself but was involved in every decision that was made throughout the course of the research. He was particularly involved in the adaptation of the research lens and the survey design and analysis phases of the research.

- **Eirini Kalliamvakou** - Post-doctoral researcher. She became involved in the project in the latter stages of the research. After the systematic mapping study had already been conducted and she had participated in our first round of pilot study for the survey, she became involved in the survey design and remained involved in the research from that point forward. She conducted some member-checking tasks for the research, but primarily acted as a consulting researcher and helped me to make appropriate decisions about the survey and its analysis and reporting.

- **Margaret-Anne Storey** - Master's supervisor, Professor. She was involved in all phases of the research project as my supervisor and acted as a consulting researcher who helped to make rigorous decisions during the research process, as well as to help direct the research project in a way that would generate meaningful insights and have an impact on the software engineering research community.

This collaborative environment helped to ensure that all of the decisions made throughout the research process were carefully considered and discussed before taking action to mitigate sources of individual researcher bias. However, for pragmatic reasons, it was not possible to involve the other research team members in more research tasks, which would have further limited possible researcher bias on the work.

Chapter 3

Background

A foundational part of this research is a framework from social science, Runkel and McGrath's models of data sources and research strategies [33]. In the following sections, I motivate why their approach to classifying and understanding research from a social perspective is highly relevant to the field of software engineering.

3.1 Why is it important to study social factors in software engineering?

Software development is a highly complex and technical process, and developers utilize a number of different technologies to design, develop, deploy, and maintain software. There is no question that technical research in software engineering has helped to advance the state of the art in software development through the creation of newer and better tools, techniques, algorithms, and approaches. However, there are a number of reasons why it is important for the software engineering research community to study the social factors that affect software development.

3.1.1 Social factors have always impacted software development

Software engineering emerged as a sub-discipline of both computer science and engineering in the 1950's and 60's; research disciplines that typically do not address the complexities of human behavior and focus on technical, logical, and mathematical problems. The first conference on software engineering was held in 1968 and

attended by over 50 people who discussed various issues surrounding the design, production, distribution, and maintenance of software [28]. While most of the discussions were highly technical, even the first conference discussed social factors that influence software development. The report on this conference outlined discussions of "the difficulties of meeting schedules and specifications on large software projects", and "the education of software (or data systems) engineers", showing that even in the early years of software engineering, experts agreed that we need to address social as well as technical factors in order to improve the process of software development.

Despite the fact that software engineering emerged as a sub-discipline of two highly technical research domains, a number of pioneering researchers produced groundbreaking work to address human and social factors of software development in the early days of software engineering.

In 1971, Gerald Weinberg published his book *The Psychology of Computer Programming* [56], which is one of the first books that discussed programming in terms of developers. Weinberg drew attention to programmers' "intelligence, skill, teamwork, and problem-solving power". Fred Brooks published the highly influential book *The Mythical Man-Month* in 1975 [6]. Rather than presenting a series of studies, Brooks drew attention to the importance of management practices in complex software projects by drawing on his own experiences and software engineering knowledge. Ben Shneiderman published his book, *Software Psychology: Human Factors in Computer and Information Systems* [40] in 1980, drawing attention to the impact of different human factors on software development. Tom DeMarco and Timothy Lister's 1987 work *Peopleware: Productive Projects and Teams* argued that the major issues in software development are social factors, and they drew on personal experiences and examples to demonstrate the impact that a good (or bad) social context can have for a software development project.

While these are only a few of the many examples of early social research in software engineering, they are highly cited within the research community. This demonstrates not only that they had a significant impact on the software engineering community, but that researchers have been working to understand the social factors of software development since the early years of software engineering as a research domain.

3.1.2 Software development is a sociotechnical system

One reason it is so important to address social factors is that software engineering involves the study of a number of **sociotechnical** systems. Whitworth described a sociotechnical system as a "social system built upon a technical base" [57]. He described four levels that make up a sociotechnical system, pictured in Figure 3.1. In this model, each additional layer encompasses the previous level to create a new system. The first level of a sociotechnical system is hardware, which makes up our computer systems and physical infrastructure. This interacts with the second level, software, to make complete technological systems. At the next level, the human-computer interaction level, individual humans interact with these software systems, adding complexity through the need to study human behavior in interactions with technology. Finally, we reach the sociotechnical level, where humans interact with each other in these technical contexts.

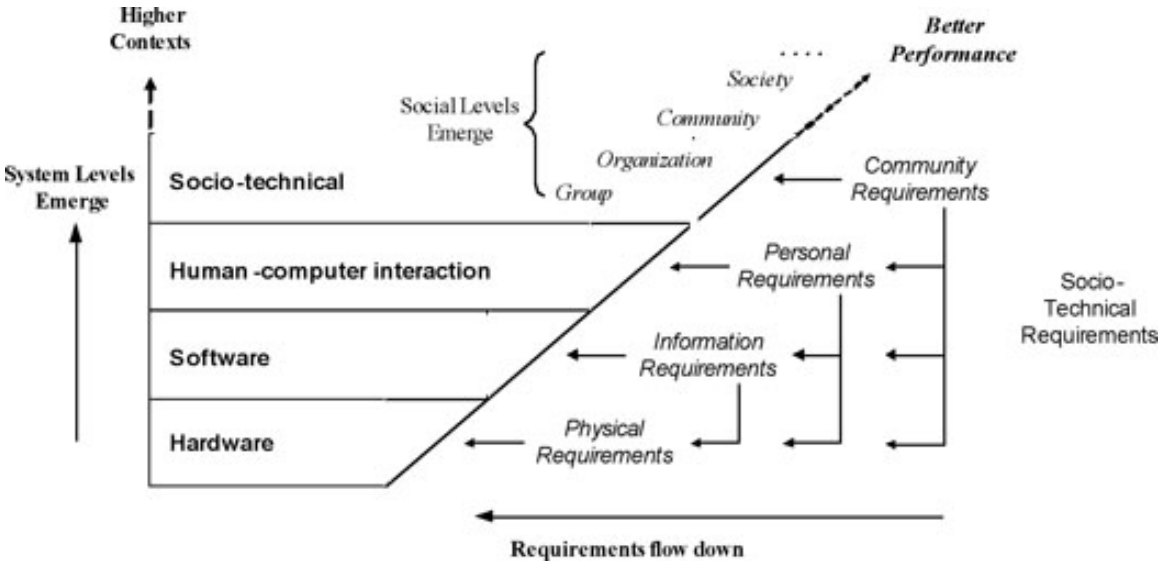


Figure 3.1: The sociotechnical model [57].

Software development is an example of a sociotechnical system. Software tools and programs run on our hardware. Developers create, maintain, deploy, and interact with software. Finally, developers engage in social activities in the larger social contexts of development teams, departments, companies, and organizations. At each level of the sociotechnical model, there are advances that we can make as software engineering researchers to help to advance and improve the process of software development. At the hardware level, we can create more efficient computers or hardware

configurations that enable software to run more quickly (though this is primarily the concern of computer and electrical engineers). At the software level, we can create new tools and software programs that help to improve a software development task, such as automated bug detection techniques. At the human-computer interaction level, we can study how developers interact with the technologies at their disposal to develop theories and approaches for software development at the developer level, informing better educational and training practices. At the sociotechnical level, we can study teams and organizations to understand the social factors that influence software development. This allows us to develop theories and approaches that can help organizations and practitioners to improve the social processes that may be adversely affecting them.

It is important to study sociotechnical systems at all levels; the most limiting factor to the efficiency of a software project could be an inefficient software tool just as easily as it could be a social context that isn't conducive to developer productivity. We can continue to make improvements through the continuous advancement of technical systems, but addressing a poor social practice could make significantly more of an impact in some organizations than improvements in the efficiency of their tools.

3.1.3 Empirical research can provide us with valuable insights

Research conducted using empirical methods "consists of gathering information on the basis of systematic observation and experiment, rather than deductive logic or mathematics" [45]. This approach to research originated in the medical community, but empirical methods have a number of benefits for software engineering [20] as they provide insights that help us move forward as a community. Basili, in his reflection on human experimentation in software engineering, pointed out that we need empirical methods in order to build foundational knowledge in software engineering. He explained that "we need research that helps establish a scientific and engineering basis for the software engineering field," [3] that it was necessary to study humans and social contexts in software development in order to develop this knowledge. He said that "the goal is to develop the conceptual scientific foundations of software engineering upon which future researchers can build" [3]. He also pointed out the benefits of using human experiments in software engineering to help build this foundational knowledge, as it helps us to understand causal factors of human behavior in software

development [4]. Experiments aren't the only type of empirical research that can help us to understand human behavior; Sharp, Dittrich, and de Souza [37] call for more ethnographic studies in software engineering, pointing to their potential to show what developers do in practice as well as *why* they do it that way.

Research conducted using empirical methods also has more potential for industry relevance. Kitchenham [18] called for the use of more case studies and quasi-experiments in industry, using methods from social science, as they would make our findings more relevant to practitioners. Empirical methods can also help us to gain a better understanding of developers, helping us to create better technologies. Sharp [38] discussed how there are a variety of both quantitative and qualitative research methods that can be used to study developers, and that the insights that can be gained from work with human participants are substantial. Sjøberg, Dybå, and Jørgensen [45] argue that doing more empirical work in software engineering will provide us with the knowledge needed to develop better technologies for software development.

3.1.4 Summary

The software engineering research community has always seen the value in understanding social factors of software development; as a sociotechnical system, software engineering is affected by human behavior at the developer, team, and organizational level. Understanding these human aspects of software development is important for building foundational knowledge causal factors of software development behavior, as well as for understanding developers and their challenges. Investigating social aspects through empirical research methods has a number of benefits for the software engineering research community; it helps us to understand the causes of developer behaviors, understand the needs of developers, make our research more relevant and impactful to industry practitioners, and gives us a foundation on which to develop better tools and technologies.

3.2 How do we study social factors in software engineering?

Studying social factors in software engineering is important, but it is also important that we discuss the methods we use to study social phenomena. As McGrath stated, in order to *"understand empirical evidence, its meaning, and its limitations, requires*

that you understand the concepts and techniques on which that evidence is based." [27]. In order to study these complex sociotechnical systems, researchers in software engineering must employ a wide variety of techniques from a number of interdisciplinary fields. To that end, there are a number of seminal works that provide guidance on state of the art techniques for empirical research in software engineering.

3.2.1 Social science influence

Software engineering is a relatively new research domain, and thus a number of research methods used in software engineering originated in other research areas. For research that addresses social factors, we borrow research methods from the social sciences, such as psychology, nursing, sociology, and education. There are a number of guidebooks from these domains that we use to investigate social issues.

The books of John W. Creswell [11, 10] are a common reference for software engineering researchers, as they provide a high-level overview on designing qualitative, quantitative, and mixed-methods research and the benefits and drawbacks of different approaches. These references are not specific to software engineering, but are more generic and apply well to many different social research domains. Mackay and Fayard provide an excellent resource and framework for understanding triangulation across disciplines [26], which is common in software engineering as it often overlaps with domains such as education, human-computer interaction, and information systems.

There are also a number of works available specifically for qualitative research. For example, O'Reilly and Parker's 2012 paper [29] provides guidance to help understand saturation and sample sizes in qualitative work. Grounded theory is a common qualitative methodology in software engineering; typically, researchers use either Strauss and Glaser's grounded theory books [24] or Charmaz's works [7] to inform their grounded theory research designs, both originating in the social sciences. For case study research, Yin's works from social science are prominent [62].

3.2.2 Making it our own

In our continued maturation as a research community, researchers have also produced a number of works that provide guidance specifically tailored to conducting software engineering research. One early example is the book *Empirical Methods and Studies in Software Engineering* [9], published in 2003. It includes a number of chapters that provide community members as well as reflections on empirical research in software

engineering. For example, it contains a chapter on how to increase the realism of experiments [46] and an introduction of four major empirical methods: "controlled experiments, case studies, surveys, and post-mortem analyses" [60].

The most prominent software engineering research book, published in 2007, is the Guide to Advanced Empirical Software Engineering [41]. This work provides software engineering researchers with guidelines for a number of different research techniques. A number of experts contributed chapters to the book based on their skillset. This book includes guidance a number of specific techniques, including qualitative methods [35], focus groups [23], personal opinion surveys [21], and data collection techniques for field studies [43]. It also provides guidance on some more general topics, such as understanding how to design ethical studies involving humans in software engineering [54] and a guide for building theories in software engineering [47]. This book also contains a chapter explaining the benefits and drawbacks of different empirical methods in software engineering to assist in research design choices [12].

Wohlin and Aurum [59] published a paper to help software engineering researchers design their studies with their decision-making framework.

We have also created our own guidebooks that adapt some social science research methods to the software engineering domain. Stol, Ralph, and Fitzgerald provide guidelines for grounded theory specifically in the context of software engineering [51]. Runeson and Høst adapt case study research guidelines to the software engineering domain [32]. Kitchenham *et. al* [19] provide guidelines for conducting systematic literature reviews in software engineering, and Usman *et. al.* provide guidelines for the development of software engineering taxonomies [53].

There are also a number of seminal works available that focus around experimentation and evaluations. Both Wohlin *et. al.* [61] and Ko, Latoza, and Burnett [22] provide excellent resources for understanding how to conduct software engineering experiments with human participants. Blackburn *et. al.* [5], explain how to assess empirical evaluations in software engineering, providing researchers with guidelines for understanding how to conduct their evaluations rigorously.

These works all help to provide a wealth of resources to assist software engineering researchers in conducting empirical research using a diverse set of research methodologies, methods, and techniques. However, they are not a complete set, and there is always room for improvement in the amount and diversity of resources available to help software engineering researchers design and conduct their studies.

3.3 Critical reflections on empirical research in the software engineering research community

Despite the breadth and depth of resources available to help software engineering researchers conduct empirical work, there are a number of software engineering studies and papers that suggest that there is still room for improvement in the quality, quantity, and diversity of empirical research we produce as a community.

3.3.1 Critical reflections on our applications of research techniques

There are a number of research papers that critically reflect on our use of specific techniques in software engineering, pointing out ways that the community can improve in its application of those techniques.

Stol, Ralph, and Fitzgerald [51] published a review of grounded theory papers in software engineering. They found that very few of the papers actually reported on their methodology in detail, which is crucial for determining rigor in qualitative work. They then discussed ways to improve our reporting of grounded theory papers and provide guidelines for the community. Similarly, Usman *et. al.*'s systematic mapping study of taxonomies in software engineering [53] showed issues with reporting procedures in the creation of taxonomies. They also presented a new framework for developing future taxonomies to help increase the quality of this work in the future.

Kitchenham *et. al.* conducted a systematic literature review of software engineering literature reviews [21], finding that the systematic literature reviews they evaluated were of fair quality but limited in their scope of topics. Their paper also provided an in-depth description of their research protocol, providing an example for other researchers to follow, as well as a call for researchers to employ the use of systematic literature review procedures in their work rather than more informal methods of literature review.

Sjøberg *et. al.* [48] published a literature review of controlled experiments in software engineering, discussing the topics, tasks, samples, etc. of the experiments reported in their sample. They found that the vast majority included student participants rather than developers and that reporting of experimental procedures was poor. They called for researchers to report on their methodologies more systematically and suggested that the high use of student developers might call into question the appli-

cability of results to an industrial setting. Salman, Misirli, and Juristo [34] tested the hypothesis that the two groups performed differently, and found that professional developers do perform differently compared to student developers in some experimental contexts, calling into question the applicability of the results of a number of laboratory experiments in software engineering.

These papers demonstrate that we can always improve in our application of empirical research methods and that a critical reflection on our research output can be very impactful; it allows for experts in the community to show us how we improve our work and produce studies of a higher quality in the future.

3.3.2 Investigations into the content of our collective research output

In addition to investigations focused on specific techniques, a number of researchers have conducted reviews of software engineering literature in a more general way to understand some aspects of our collective body of work.

Shaw [39] investigated the papers submitted to ICSE 2002, analyzing the content of the papers that were both accepted and rejected for the conference, as well as observing program committee conversations about what papers to accept. She found that there were very low rates of submission and acceptance of papers that investigated "categorization" or "exploration" research questions, or papers whose research results presented "qualitative or descriptive models". This research provided an in-depth understanding of the types of questions that were addressed by ICSE papers, as well as what the program committees were looking for in papers that they accepted for the conference; this means that her work provided guidelines for how to *write* better research papers in software engineering for increased acceptance, but not necessarily how to design better quality studies. A 2016 replication of this methodology [52] showed that since then, reviewers have raised their standards, particularly with regards to empirical evaluations of research contributions. This is a good sign that empirical research is increasingly prominent in software engineering. The replication study also found that a new category of research papers, mining software repositories, was incredibly common. This shows how quickly the software engineering community can adapt to the use of new research techniques and technologies.

Siegmund, Siegmund, and Apel [42] investigated the research community's understanding of the tradeoffs between internal and external validity in empirical software

engineering research through a literature review as well as a survey of reviewers. They found that there was an inconsistent discussion in the literature about threats to internal and external validity in empirical studies, as well as a lack of consensus among reviewers about which type of validity should be prioritized in a particular study. Zelkowitz's work [63] found that the community's use of empirical validation techniques for research contributions was improving; they found significantly more case studies, field studies, and dynamic analysis techniques than previous decades. They also found that some problems mentioned in previous studies (such as a lack of access to software repositories) may no longer be an issue. Unfortunately, they noted that researchers were using terms such as "case study" to refer to different levels of abstraction, which makes it more difficult to understand the way this research is communicated.

Høfer and Tichy [13] investigated 10 years of empirical software engineering research, from 1996 to 2006, finding that the majority of papers presented experiments or case studies, that the primary topics of study were metrics and tools/frameworks, and that a number of important topics were underrepresented. This pointed to a narrowed scope of empirical research topics compared to the broader spectrum of software engineering research as a whole, and they called for empirical researchers to broaden their horizons into different topic areas.

3.3.3 Calls to action from the community

Software engineering researchers have published a number of position papers over the years showing the benefits of empirical research and active human participation in studies. These papers present a collective "call to action", suggesting that the community should diversify their research methods to produce more varied empirical work.

One thing community members are calling for in particular is for us to conduct more realistic experiments and quasi-experiments with developers in software engineering. Basili drew attention to the importance of experimentation and empirical research in software engineering, discussing the benefits and drawbacks of controlled experiments. He explained that this type of empirical work can be used to develop knowledge into the causality of different human behaviors in software engineering, helping to develop foundational knowledge about the process of software development [3, 4].

Kitchenham's paper [18] pointed out that highly controlled laboratory experiments were perhaps too heavily emphasized, and that they might not reflect the reality of software development. She called for the use of more case studies and quasi-experiments in industry, using methods from social science, as they would make our findings more relevant to practitioners. Sjøberg *et. al.* [44] suggested not only that software engineering researchers should conduct human experiments, but that they should strive for these experiments to be as realistic as possible; they called for community members to apply for resources in order to conduct highly realistic software engineering experiments. They suggested that conducting these types of resource-intensive experiments would help to show the value and validity of the findings to industry practitioners, helping to make research more impactful in real-world software development.

Researchers are also calling for more empirical research that addresses social factors of software development without focusing on experimental approaches. Sharp, Dittrich, and de Souza [37] call for more researchers to conduct ethnographic studies in software engineering, pointing to their potential to show what developers do in practice as well as *why* they do it that way. Sharp also [38] outlined the importance of understanding social and human aspects of software engineering. She discussed how there are a variety of both quantitative and qualitative research methods that can be used to study developers, and that the insights that can be gained from work with human participants are substantial.

Sjøberg, Dybå, and Jørgensen [45] also drew attention to the importance of human factors when they presented their vision for the future of software engineering. They argued that doing more empirical work in software engineering will provide us with the knowledge needed to develop better technologies for software development. They also suggest a number of possible ways to increase empirical method use in software engineering: increased education and experience in a variety of empirical techniques, better industry cooperation, alignment of industry and academic priorities, and additional resources for empirical work.

The above position papers, through their various messages, present a unified call to action. Empirical research methods have a number of benefits for software engineering as a whole and addressing social factors of software development in field settings will help us to move forward as a community by providing us with a sufficient knowledge base to guide our development of new tools, approaches, and frameworks for software development.

3.3.4 Summary

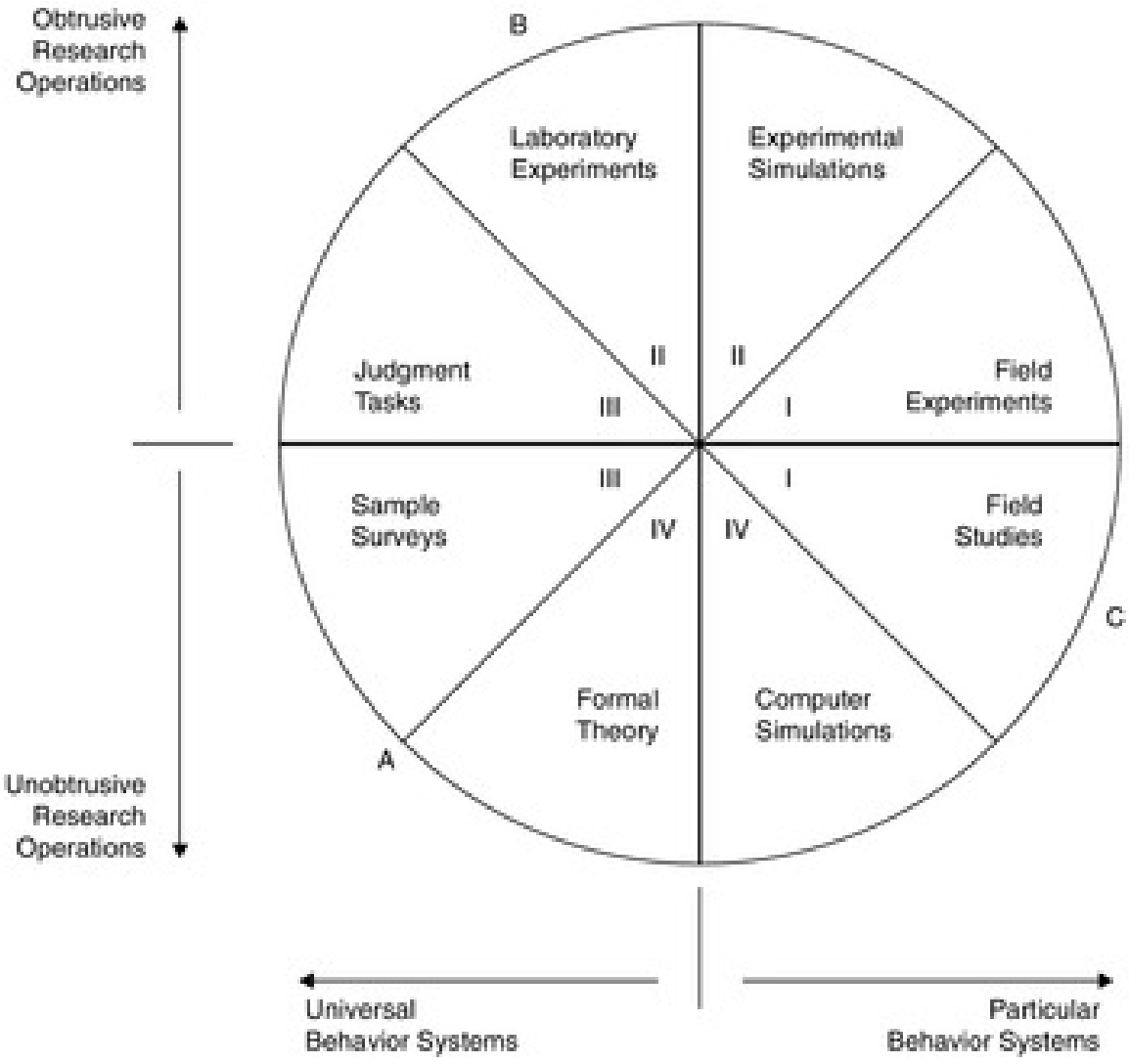
Overall, these papers show that critical reflections and meta-research in software engineering help us to understand our weaknesses, and provide a platform for discussion around the research we conduct in software engineering. They allow us to continuously raise our standards for research quality by forcing us to reflect on our work and providing us with possible ways to improve. I suggest that while position papers and meta-research studies do not provide a "solution" to a software engineering problem, they are critical to the progression of software engineering research because critical reflections on our practices stimulate discussion about improving our techniques, which in turn improves the research we produce.

3.4 Understanding software engineering research from a sociotechnical perspective

One way to understand software engineering research from a sociotechnical perspective is to use Runkel and McGrath's model of research strategies.

Runkel and McGrath's 1972 work *Research on Human Behavior: A Systematic Guide to Method* [33] and McGrath's follow up paper "Methodology Matters: Doing Research in the Behavioral and Social Sciences" in *Human-computer Interaction* [27] both present a series of models that help to explain the benefits and drawbacks of various research design choices in the context of human behavior research. These models were created outside of the domain of software engineering and were designed to apply to human behavioral research as a whole. Techniques for manipulating variables, different sources of data, the different types of threats to validity, data comparison techniques, and sampling strategies are just some of the topics touched on in their works. However, these works are best known for discussing Runkel and McGrath's "circumplex" of research strategies, shown in Figure 3.2. It was introduced first in the 1972 book and adapted slightly for wording in the 1994 paper [27]. The circumplex model of research strategies allows for researchers to understand how the choice of what high-level approach to take in a study can affect different aspects of quality in the resulting work.

Since the inclusion of McGrath's paper in *Human-computer Interaction*, this model has been cited in a number of various sociotechnical research domains, including human-computer interaction [8, 17], software engineering [25, 55], and information



I. Settings in natural systems.
II. Contrived and created settings.
III. Behavior not setting dependent.
IV. No observation of behavior required.

A. Point of maximum concern with generality over actors.
B. Point of maximum concern with precision of measurement of behavior.
C. Point of maximum concern with system character of context.

Figure 3.2: Runkel and McGrath's circumplex of research strategies [33].

visualization [36, 15]. However, none of these works attempt to classify and understand the software engineering research community through the use of the circumplex; the circumplex model helps to support their work but is not the primary focus of the studies.

So far, only one research team has applied the circumplex model to learn about the software engineering research community; Stol and Fitzgerald's 2015 paper "A Holistic Overview of Software Engineering Research Strategies" [50] describes this model in the context in software engineering. Stol and Fitzgerald make the case that there is a lot of confusion surrounding how to discuss research and that the circumplex model would be a useful way to approach solving this problem. They discuss how there is no shared taxonomy that allows for a common understanding of what terms such as "case study" really mean, and that discussing research at a higher level, the research strategy, would help to alleviate this confusion. They also suggest that the use of the circumplex model in software engineering would allow the research community to make a connection between the choice of overarching strategy for a study its impact on aspects of research quality. Stol and Fitzgerald suggest that understanding this connection would allow community members to have more rigorous discussions when reviewing papers for publication, and when conducting meta-research. The paper then goes on to explain Stol and Fitzgerald's interpretation of the circumplex model and show examples from software engineering literature that contextualize what each research strategy looks like when it is applied to software engineering. While they explain the circumplex very well, it is just *one interpretation* of the model and there are some limitations to the work. These issues and limitations will be discussed further in Chapter 8.

Chapter 4

Research Lens

In order to critically examine and accept the findings from this research, it is crucial to first understand the models that shaped the research process and are used to communicate the findings. This forms the **research lens**, the models that formed the theoretical grounding of my work. Our research lens consists of adaptations of Runkel and McGrath's models of research strategies and data collection methods[33]. These models were originally developed and described in the context of the traditional social sciences, like psychology and sociology, and not for sociotechnical domains like software engineering or human-computer interaction. The addition of technical aspects quickly introduces additional fringe cases that are not easily categorized in the model without additional information. Additionally, the model was created in 1972, long before the creation of many technological phenomena such as virtual work environments, video calling, and social media that can complicate what we consider a setting, a behaviour, or an actor. This meant that we needed to extend the models in order to describe how modern technology affects the definitions of each category, as well as contextualize these definitions by using examples from the software engineering domain. Moving forward with this work, I use the term **research lens** to refer to the adaptations of Runkel and McGrath's models described below.

4.1 Research Strategies

First, we extended Runkel and McGrath's model of research strategies to the software engineering domain. The "circumplex" places eight different research strategies in segments of a circle, each separated into four quadrants containing two research

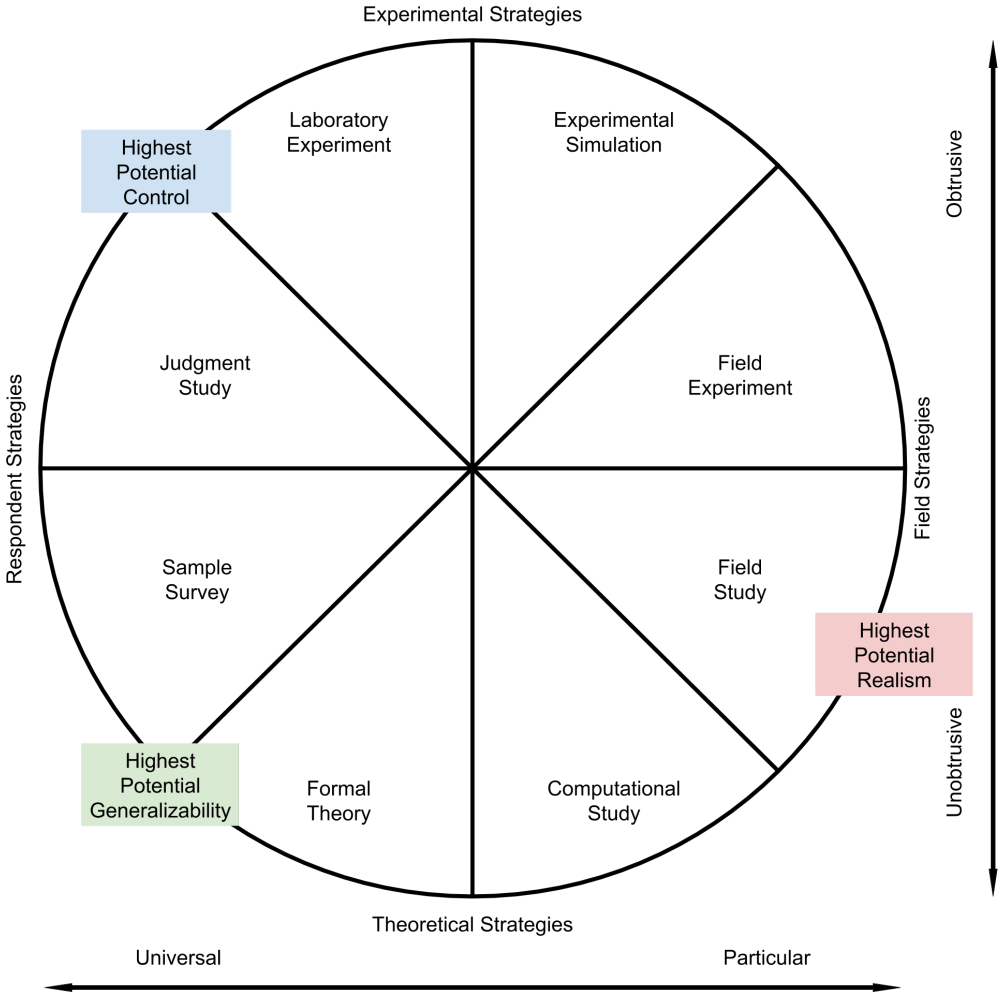


Figure 4.1: The Circumplex of Research Strategies, adapted for the software engineering domain

strategies each, as seen in Figure 4.1. Each quadrant is distinctly different in terms of the *setting* where the research takes place. In the first quadrant are the Field Strategies, which describes research that occurs in natural environments. The second quadrant contains the Experimental Strategies, which describes behavioural research that is conducted in contrived environments created for the purposes of research. The third quadrant has the Respondent Strategies, where research is conducted in such a way that the participant's surroundings are made as irrelevant as possible to the research. The fourth quadrant refers to Theoretical Strategies, where no new observations of behaviour are being collected and the primary research instruments are computers and researcher cognition. The circumplex can also be described in terms of axes; along the horizontal axis, the strategies furthest to the left of the circumplex are the most universal behaviour systems, while those furthest to the right are the more particular and context-specific. Along the vertical axis, strategies at the top of the circumplex are highly intrusive, while those at the bottom of the circumplex are quite unobtrusive and are not very disruptive of natural behaviours or settings. Runkel and McGrath also labelled points along the circumplex where three desirable research criteria are at their highest potential for maximization, as described in the following section.

4.1.1 The Desirable Research Criteria

The most important part of this model for our analysis are the three desirable research criteria because they allow us to make a connection between the research strategies used in software engineering and their potential impact on research quality.

The first criterion is ***control*** *of measurement of behaviour over extraneous factors.* This describes the level of control the researcher has in isolating and controlling extraneous factors in research, which determines the degree of confidence the researcher has in knowing which factors caused the observed behaviour under study. Runkel and McGrath also interchangeably refer to this as "precision"; for clarity, we only refer to this as control. In research that is high in control, it is less likely that the behaviour the researcher is observing is caused or influenced by anything other than the variables for which the research team has controlled. On the other hand, a study that is very low in control observes behaviour that is occurring without any external controls over the situation; the researcher can record contextual information, but without hav-

ing controlled for extraneous factors they are unaware of which contextual factors, actions, or perceptions caused or influenced the behaviour they have measured.

The second criteria is ***generalizability*** *over a population of actors*. It is not possible to conduct research on an entire population; research must be conducted using a subset of a population with the aim that the findings are applicable to the remainder of the population. Research that is highly generalizable means that the findings could be applied to virtually all of the actors within the larger group, whereas research that is low in generalizability is highly context-specific and not easily applied to a larger group of actors.

The third criteria is ***realism*** *of context*. This refers to the extent to which the context under which the data was gathered reflects real life, and to which the findings of the study are true to real-world behaviour systems. Research that is highly realistic is concrete and the findings are easily transferred to real life, and normally this is because the research was conducted on natural behaviours in real settings. Research that is low in realism, on the other hand, tends to be conducted in contrived scenarios that may not reflect the reality under which the behaviours under study occur naturally.

There is an inherent tradeoff that must be made between each of these criteria. It is not possible to have a single study that is high in control, generalizability, and realism simultaneously. This means that researchers must choose studies that have the inherent weakness of having low potential to maximize some criteria to be higher in others. In order to solve this problem, researchers must triangulate across the circumplex, using complementary strategies that make up for each other's weaknesses. This means that researchers can create a collective body of work that is high in generalizability, realism, and control by conducting multiple studies that investigate the same phenomenon using complementary research strategies.

4.1.2 Field Strategies

Field Strategies in software engineering involve researchers entering the natural setting of the actors under study in order to conduct their research. This can be a physical environment, like a software development company, or a virtual environment, like the communication channels used by a distributed development team. Because the behaviour and contexts under study are naturally occurring, field strategies have a high potential to maximize *realism of context* but are very low in potential *control over*

extraneous factors. Findings tend to be contextualized to specific environments and behaviours, so it can be difficult to *generalize* to a larger population. The distinction between a *Field Study* and a *Field Experiment* is the degree of *control* the researcher exercises in the situation under study.

In **Field Studies**, the researcher does not manipulate the natural setting under study. Observational studies and case studies are common types of Field Studies in software engineering. One example of a Field Study is where a researcher visits a company that follows agile development practices and observes how these practices affect what constitutes effective management styles. These studies are fairly *unobtrusive* because they allow natural behaviour to occur without intervention.

Field Experiments differ from Field Studies by introducing a controlled condition into the situation under study to understand the effects it creates, compromising some *unobtrusiveness* and potential *realism* for higher potential *control*. Field Experiments may be less common than Field Studies because interventions within the development process can introduce potential ethical concerns or a risk of lowered productivity that industry participants may be unwilling to accept. One example of a software engineering Field Experiment would be introducing a novel automatic testing tool to a company's development process and observing its effects on code quality.

4.1.3 Experimental Strategies

Experimental Strategies in software engineering involve testing hypotheses in highly controlled situations. These strategies yield high potential *control* but at the cost of *reduced potential realism of context* and *narrowed potential for generalizability*. The difference between a *Laboratory Experiment* and an *Experimental Simulation* is whether or not the context that is created for participants is kept intentionally simplistic and sterile, or meant to replicate a specific natural context.

Laboratory Experiments refer to situations created by the researchers where individual participants or groups take part in an experiment. The context is kept deliberately generic for all participants, meant to remove many contextual elements that exist in natural behavior settings that could influence participant behavior. This strategy is used when researchers *focus on a certain behaviour* and wish to measure it with *considerable control*. For example, a researcher conducting a Laboratory Experiment on a debugging tool may invite graduate students to a lab and ask them

to accomplish a set of *predetermined* debugging tasks with and without the tool using a simple, generic computer setup in a relatively empty and quiet room.

Experimental Simulations in software engineering aim to *replicate some aspect of the participant's natural environment* during a controlled experiment, thus *gaining some potential for realism*. For example, a researcher investigating project management meetings may conduct an experiment in a room with a similar setup to the one used at the company where actors simulate a typical meeting for the participants.

4.1.4 Respondent Strategies

Respondent Strategies are used to systematically gather participant responses to questions posed by the researcher. The main difference between *Sample Surveys* and *Judgment Studies* respectively is whether the study aims to gather *information about human perception or behaviour under a stimulus* (i.e., respondent attributes), or *information about the stimulus itself*. These strategies aim to make the participant's physical setting and conditions irrelevant to their responses. Respondent Strategies can be more convenient than Field Strategies because they often do not require physical access to an industrial environment and can be remotely conducted.

Sample Surveys are used to investigate the effects that a phenomenon has on human behaviour by surveying specific members of a chosen population, often aiming at generalizing the findings to more of the population. For example, a researcher aiming to improve continuous integration tools may distribute an online survey, asking developers to describe how they use these tools and what challenges they face. They are not limited to "surveys" in the traditional sense; this strategy can employ interviews, focus groups, or other means of soliciting opinions in addition to the traditional questionnaire. Sample Surveys are high in *potential generalizability* because they are typically conducted with a group of participants that is *representative* of the larger population.

Judgment Studies are commonly used in software engineering to evaluate the performance or utility of a new tool or technique, as they use participant perceptions and behaviours to learn about some stimulus that is applied to the population under study. Judgment Studies tend to be *high in potential for control of measurement* of both the stimulus materials and the responses; however, they are often *lower on potential for generalizability* over the population compared to a Sample Survey, as they are typically done with "actors of convenience" or relatively small population

samples. In the context of software engineering, this normally means sending a tool to the developers whose software helped to inform the creation of the tool.

4.1.5 Theoretical Strategies

Theoretical strategies differ from the previously described strategies because they are *the only methods that do not involve the inclusion of active human participation as part of the research* (but the studies may be based on past empirical data and studies). They are low in *potential control* because they are not conducting new observations of human behaviour, and thus they cannot easily control for extraneous factors.

Computational Studies refer to computer experiments that have a *complete and closed system* to model operations without any human involvement or dynamic feedback from the outside world. The primary tool of the researcher is a computer. These studies are very common in software engineering and can be conducted using a wide variety of techniques, including experiments to evaluate software tools, data mining studies, computational analysis of big data, the creation and evaluation of prediction models, natural language processing techniques, and computer simulations. This strategy was originally named *Computer Simulation*, but we changed the name to Computational Study to reflect the varied nature of studies conducted using this strategy in software engineering. For example, a researcher aiming to evaluate a new bug detection technique may use version control history in an open-source project to see if their tool identified all the bugs that were fixed in subsequent versions of the project. Another example is running a series of experiments comparing the performance of various state-of-the-art static Android security analysis tools. Computational Studies may use methods for gathering and analyzing digitized data, which is common in data mining studies.

Formal Theory research does not involve gathering new empirical data but rather focuses on the creation of models and theories based on previously gathered data or existing theories and models. In a Formal Theory study, the primary tool of the researcher is their own mind [27]. Like Computational Studies, there are many different types of Formal Theory studies in software engineering, including, but not limited to, qualitative synthesis studies, literature reviews, the creation of a theory from past empirical work, and purely mathematical or logical research papers. For example, by building on previous models, a theory formulation study may aim to identify and describe underlying factors that can explain why certain practices

support alignment and coordination in software projects. These theories are meant to be context-independent, so Formal Theory work is *high in potential generalizability.*

4.2 Data Sources

In addition to research strategies, McGrath [27] also describes a number of possible empirical data sources for behavioural research, and the benefits and drawbacks of each. These sources help us to determine the level of human involvement in the research.

Self Reports refer to instances where participants voluntarily report on their own behaviour or perceptions for research purposes. In software engineering, this usually means responding to direct researcher questions through a questionnaire or an interview. They have the benefit of being able to determine a participant's perceptions about a topic from their own perspective, but they have the drawback of being subject to certain biases; a participant might not want to respond to a question in a way that would reflect badly upon them, which can influence their responses.

Observations by a Visible Observer and **Observations by a Hidden Observer** are observations of human participants; either participants are aware they are being observed or measured (Visible Observer) or they are unaware (Hidden Observer). Data gathered through these methods occurs in real time, and does not include data about behaviour that occurred prior to the start of research. Observational data can come from a variety of techniques, including sensor data, video and audio recordings, one-way mirrors, and being physically present in the same room as a participant. Observational data has the advantage of being able to show how participants respond to different stimuli in real-time, potentially at the control of the researcher. As a disadvantage, observational data can be prone to error, either via the researcher's perception or through the fault of some measurement device. Visible Observer data is also highly "reactive", meaning that the data may be influenced by the participants *reacting* to the fact that they are being observed, potentially changing their behaviour to what they may believe the researcher is looking for. Hidden Observer methods do not have this limitation, but are instead plagued with the ethical concern of researching someone without their knowledge. Common examples of Visible Observer data in software engineering are recordings of code written by a participant in a coding experiment conducted in a lab, or observing a development meeting in a company. Hidden Observer data is less common in software engineering,

but one example is entering a development team and observing their behaviour for research purposes under the guise that you are a new team member.

Public Archival Records and **Private Archival Records** are data about human behaviour that is recorded by a third party for non-research purposes, but is used as the subject of research after the fact. The difference between them is that private records would be unlikely to become a matter of public record, like a diary entry. Both of these data sources are fairly uncommon in software engineering research; public records are often very easy to access and can be useful for showing trends over time. For example, university graduation and enrollment statistics can be useful for showing trends in software engineering education. Private records, on the other hand, are often very difficult to access, especially from software companies, due to security and ethical concerns. However, if researchers can get past these access barriers, these records they can be very useful in researching phenomena that have been under-explored in previous studies. For example, a software company's high-level production meeting minutes may be very difficult to access, but a researcher could analyze this data to understand how decisions about software are made by non-technical stakeholders.

Trace Measures are records indirectly created by humans as a result of their behaviour. Humans create these measures on their own; they are not collected by a third party and they are not created for the purposes of research. Most software development artifacts fall into this category as they are *traces* created by developers as a result of software development behaviour. For example, software is written by developers to fulfill some need, but later the source code (or its bugs, commits, or error logs) becomes a Trace Measure we can study in future research. There are benefits to these types of data; often, they are publicly available and thus they are very easy to access. Additionally, they are records of behaviour that was not influenced by the knowledge that the traces would be analyzed for research. However, there are drawbacks to using Trace Measures, particularly with the lack of control over measurement and lack of context available to explain such data. For example, if a researcher is looking at Trace Measures of a developer's activity on GitHub, they are limited in what they can do with that information. They are not able to make changes to that developer's life that could influence or change their GitHub activity, or record contextual information about that developer to explain why certain events or patterns occurred within these logs of their activity.

To extend McGrath's work, we also separate these data sources by the extent of human participation in the research. Self Reports and Observations are considered *active forms of human participation* because the researcher is interacting with research participants in real-time, enabling more contextual data about the participant and their behaviour or responses. Archival Records and Trace Measures are *inactive forms of human participation* because the data is collected from records of past behaviour, and thus the participant is not actively engaged in the research at the time the research is conducted. This data can often be easier to collect, but the researcher is often unable to exert control for the factors that led to the creation of the data or collect additional contextual information. Purely theoretical research does not consider empirical data, but rather focuses entirely upon theoretical constructs, such as algorithms, design patterns, and logical or mathematical proofs. In these cases, there is *no form of human participation* in the research. This category was not part of the original model and was added to denote research that does not utilize empirical data. Moving forward, this categorization is referred to as **Formal/Theoretical**.

Much like the research strategies, these data sources have their benefits and drawbacks, which can be offset by triangulating data sources. For example, if a researcher is conducting a survey relying on Self Reports from participants, that survey could potentially be influenced by the participant's desire to not be judged negatively. That data could be triangulated using Trace Measures of that participant's behaviour, which would be unbiased by their participation in the study and could corroborate the claims made by the participant. It is important to note that a researcher can use more than one data source in the same study; A study can follow only one research strategy, but that strategy can gather data from multiple data sources for the purposes of triangulation.

4.3 Fringe Cases

These adapted models reflect a rigorous and iterative process of identifying papers that challenged our interpretations and altering our descriptions of the models to accommodate these papers. However, as I applied this research lens to ICSE papers I still encountered a number of cases that were not easily classified. Throughout the remainder of this thesis, these papers are referred to as *fringe cases*. In a fringe case, the methods used in the papers could be argued as being in one of two categories depending on your perception. One such example is data from online question and answer

forums. Our interpretation of this data is that this data is a *Trace Measure* of the human behaviour of asking questions and receiving an answer. Another interpretation is that a *third party* (the website) collected these questions and answers *and publicly archived them for non-research purposes*. In this way, online question and answer forum data could be classified either as a Trace Measure or as a Public Archival Record depending on your perception. Most of these fringe cases were revealed through the survey we conducted (described in Chapter 5), as authors categorized their work and explained their reasoning in such a way that revealed a new way of thinking about their research strategy or data source. Because these fringe cases were identified after the completion of the systematic mapping study, the classifications in the systematic mapping study consistently reflect my own initial perception.

4.4 Summary of Adaptations

The descriptions above reflect a number of adaptations to the original models as described by Runkel and McGrath. First, Runkel and McGrath describe certain points in the circumplex where the generalizability, realism, and control are maximized; we instead refer to these points in terms of their *potential* to maximize those criteria. This is because we recognize that the research strategy only provides a situation where certain criteria are much easier to maximize than others and that the research team must conduct a methodologically sound study in order to actually maximize those criteria.

One major change made to the model was the renaming and adapting of the research strategy "Computer Simulation" to "Computational Study". The original model refers to a fairly simplistic definition of Computer Simulation, where it refers to simulated models of human behaviour. As the original model was made for purely behavioural research, it does not capture the complex sociotechnical reality of software engineering research. We saw many examples of this category of work that followed diverse computational research approaches to examine digitized data as part of a closed system, including data mining studies, prediction models, and experiments. As this category now represented more types of investigation than just "simulations", we renamed the category "Computational Study" to reflect the diverse nature of these studies in software engineering, all centering around the use of a computer as the primary tool of the research in a closed system.

We also made a key change to the data sources by adding a new category, "Formal/Theoretical", to refer to papers that do not include empirical data. This was not included in the original model of data sources because it is not actually a data source, but rather the lack of empirical data. This was included in order to have a means for classifying formal or algorithmic research that does not work with empirical data. We also separated the 7 data source types into active and inactive forms of human involvement, and the lack of human involvement whatsoever. Runkel and McGrath did not make this distinction, however, we believe it is useful to add these categories to demonstrate the extent to which humans are *actively* engaged in our research.

The final and most significant adaptation I made to the models was re-writing the descriptions of the models using terminology and examples from software engineering. All the examples provided in Runkel and McGrath's original descriptions come from the social sciences and humanities, so it is difficult to understand how to apply the original models to software engineering, which studies very distinct sets of actors, behaviours, and contexts using a diverse set of sociotechnical research methods and data sources. Therefore, I provided examples that contextualize each of the eight research strategies and seven data sources as they typically exist in the context of modern software engineering research to aid others not only in understanding this research, but to help them apply the models to other contexts as well.

Chapter 5

Methodology

The research described in this thesis was conducted in five phases. The first phase was a systematic mapping study, where I categorized papers according to the sociotechnical research lens described in Chapter 4. The second phase involved the design of a survey, an iterative process that involved several rounds of piloting, analyzing results and feedback, and implementing changes. The third phase of my research involved survey dissemination and data collection. The fourth phase of our methodology was the analysis of the survey and systematic mapping study together, which included both qualitative and quantitative analysis techniques. Finally, we completed member checking tasks in phase five to validate our findings.

The research questions I address are:

- **RQ1:** What research strategies and data sources are described in research published at ICSE?
- **RQ2:** Why do authors choose the research strategies and data sources they describe in their papers?
- **RQ3:** What is the "balance" of quality aimed for across generalizability, control, and realism in research published at ICSE?

5.1 Systematic Mapping Study

In the first phase of the research, I conducted a systematic mapping study [30, 39] to address RQ1 and RQ3. I considered all technical research track papers from ICSE's 2015, 2016, and 2017 proceedings in my sample. I collected 84, 101, and 68 technical

track papers from each year, respectively, for a total of 253 papers. I excluded other conference tracks and paper types.

I chose ICSE because it is widely considered the flagship conference in software engineering, and does not focus on a specific set of topics or methods necessary for inclusion in the conference. I selected three years of proceedings for two reasons. First, a program committee's makeup may influence method use or topic choice within the set of accepted papers, so three years of data would ensure that the biases of each program committee would be mitigated by the inclusion of additional years. Second, I wanted to understand the current state of software engineering research rather than to show trends over time. I elaborate on the limitations of this choice in Section 8.3.1.

Second, we developed rules to use for our classification. We iteratively refined Runkel and McGrath's descriptions of data sources and research strategies as I applied them to the ICSE papers in our sample. Finally, I classified the papers according to these completed descriptions and recorded the classification of each paper, along with the reasoning for the classification, in a spreadsheet. Following this, I collected statistical information for further analysis, described in Section 5.4.3. These preliminary results showed a high use of Computational Studies and Trace Measures over other research strategies and data sources, further described in Chapter 6. A summarization of the systematic mapping study methodology is shown in Figure 5.1.

5.2 Survey Design

The survey was designed to complement the data collected during the systematic mapping phase by addressing all three research questions. The design process included multiple pilots and an ethical approval process.

5.2.1 Design Process and Piloting

The initial survey was designed to answer my three research questions and had three sections: The first section asked demographic questions, the second asked questions about the ICSE paper authored by the participant, and the final section asked general questions about the participant's research career. The survey was implemented in Google Forms, which was chosen because of its clean interface and ease of use, and because other team members had experience with using it as a surveying tool in their previous work.

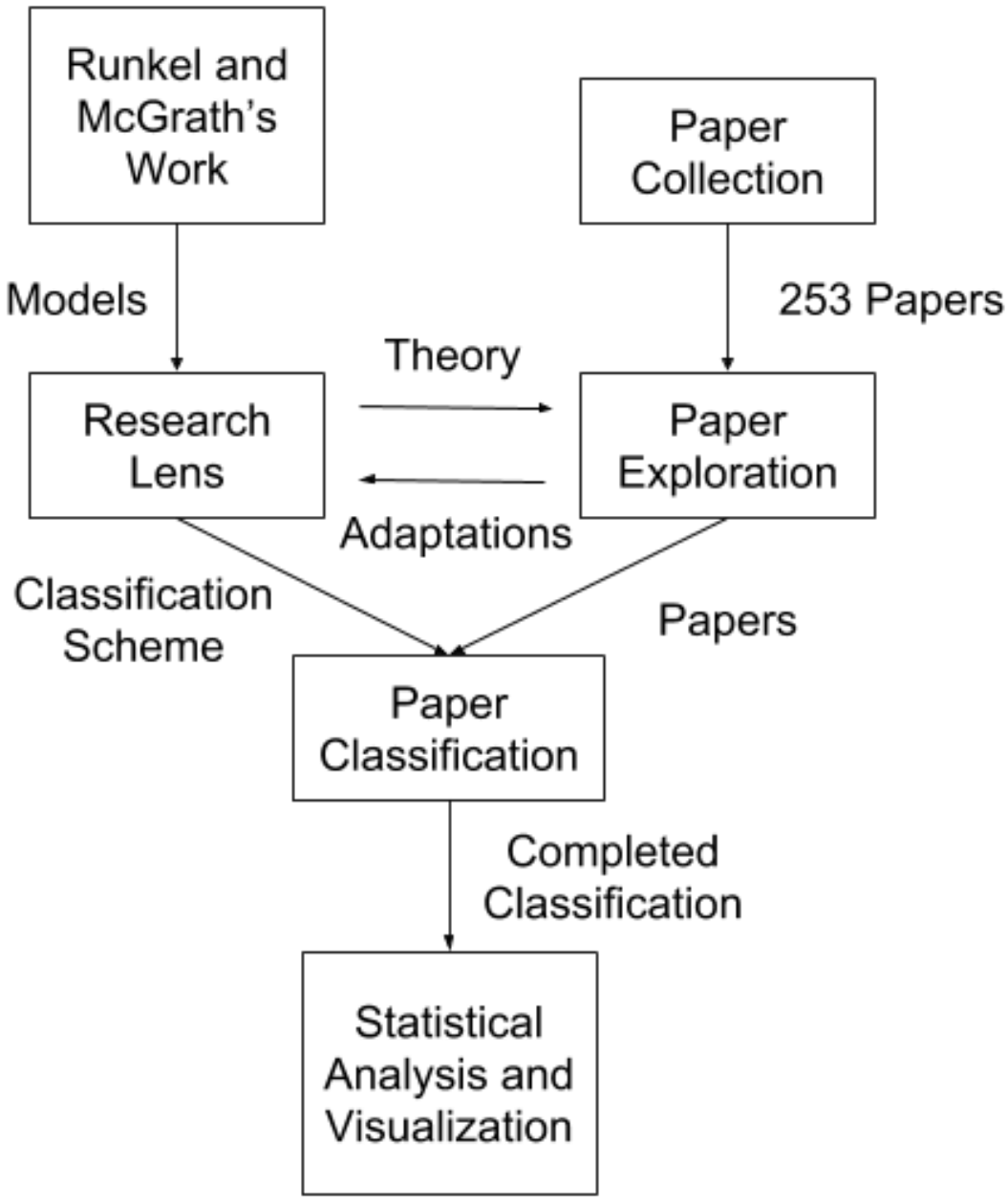


Figure 5.1: The systematic mapping study methodology.

This design was discussed by our team and edited until the first pilot. The pilot was sent to three fellow researchers; the first was a team member who was not involved with the survey design until this point. The other two participants were fellow graduate students in the lab who had conducted some unpublished studies. All three participants were asked to respond to the survey as well as take note of anything they thought was problematic. This process highlighted several issues with the survey that prompted changes. For example, there were issues with the question that asked about triangulation. We did not provide much context about what we meant by triangulation, and the participants were unsure how to answer the question. We debated whether or not to provide a definition of triangulation, as including the definition would make sure that all of the participants understood what we meant, but at the same time it could bias their responses and prevent us from gauging how well participants understood this concept in the context of research. We reached the compromise of asking participants to provide their own definition of triangulation and describe how they use triangulation in their work.

Another issue that arose from the first pilot was that the research lens was complicated and took too long to read. To fix this, I separated the definitions from the examples and made the definitions appear as an image before the response area, and kept the examples within the response area. This was done so that the participants would read the definitions, and then when responding, the examples would contextualize their interpretation of our definitions so that if they misinterpreted the lens, the examples would hopefully prompt a re-reading of the definitions.

At this point, we needed to decide whether or not to randomize the order in which the research strategies or data sources would appear. We determined that it was necessary for the definitions to appear in a set order because it aids in understanding of the lens, as adjacent options are similar and reading them one after the other highlights the differences between them. Randomizing the response order would mitigate a potential bias in responses, however, we thought it could be confusing for the responses to be in a different order than the definitions directly above the question.

To determine whether random ordering caused confusion I conducted the next pilot as a think-aloud study and selected a graduate student participant whose work I knew well so it would be easy to interpret their responses. I asked them to respond to the entire survey while verbalizing their thought process and comment on any issues they saw with the survey. This pilot showed that many of my corrections from the

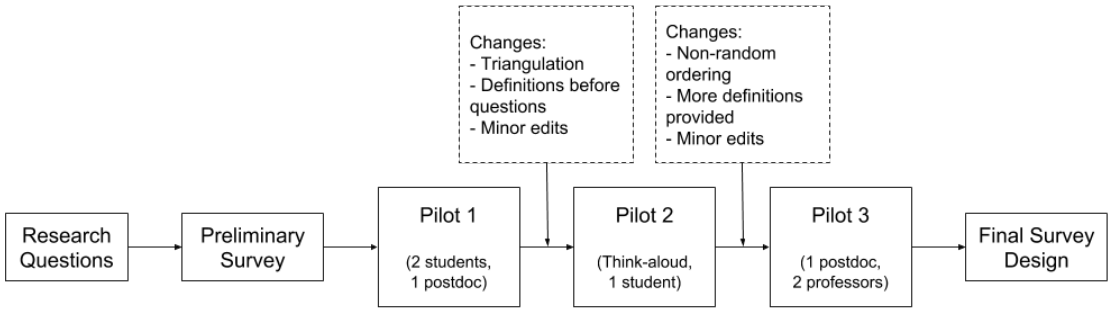


Figure 5.2: The survey design process.

first pilot were effective, that randomization of responses was ineffective and caused confusion, and highlighted some other minor issues we were able to fix.

After this pilot, the survey design was reaching completion and I began the final stage of piloting. I sent this version of the survey to three more researchers: a post-doctoral researcher within our research group and two professors within the department. There was some confusion with our terminology, however, I did not make changes to the survey as this would have jeopardized the ability of the survey to complement the systematic mapping study. The systematic mapping study classifications rely on the use of the models of research strategies and data sources, so the use of another set of terminology for the survey would have made the results of the survey and the systematic mapping study incompatible. After finishing the analysis of the responses to this pilot, I determined that the participants understood the questions and responded as we expected. The final version of the survey is included in Appendix A and the design process is summarized in Figure 5.2.

5.2.2 Survey Topics

To address RQ1, I asked the participants to classify their own papers according to the adapted research lens. This complemented my own classification of the papers, allowing us to validate the systematic mapping study and identify potential limitations. This also allowed us to then ask the authors why they made these choices, addressing RQ2.

To complement our data about generalizability, realism, and control for RQ3 I asked authors about these criteria. This was in the context of their ICSE paper,

their own priorities, and how they feel the community perceives and prioritizes these criteria.

The survey also asks some questions that do not directly support the research questions but mitigate possible limitations to validity within the survey itself. For example, to understand the limitations of choosing only ICSE papers and authors as our sample, we asked participants whether they used any research strategies that were not reported or published, about the research strategies that they used in the last five years as a whole, as well as about how they use triangulation. This would help us to gauge whether work published in ICSE was representative of typical research designs for ICSE technical track authors regardless of venue.

5.2.3 Ethical Considerations

As part of the survey design process, I submitted a Human Research Ethics Board application for approval of my research protocol. This included considerations about participant anonymity, data storage and disposal, and publication of findings. This application was submitted as a modification of the research protocol for my interview study, which was approved with the number 17-055. As part of this ethics modification, I submitted an email invitation script, an implied consent form, a copy of the survey, and the application for the modification to the existing research protocol. This package is attached in Appendix B.

5.3 Survey Dissemination and Data Collection

Our goal for survey dissemination was to get responses for as many papers as possible but to receive a single response per paper. As such, we decided to focus on contacting the first authors of the papers as they are likely the best choice for answering questions about what work was done, and particularly about why they chose those research strategies and data sources. A decision tree showing how we invited the authors to participate in the study is shown in Figure 5.3.

First, I collected the names and email addresses of the first authors from the title sections of each of the 253 papers and organized them in a Microsoft Excel spreadsheet. When an email address was not provided in the paper for the author, I used publicly available email addresses. After collecting email addresses, I sent survey invitations to the first 10 authors from ICSE 2017. This was done so that I could

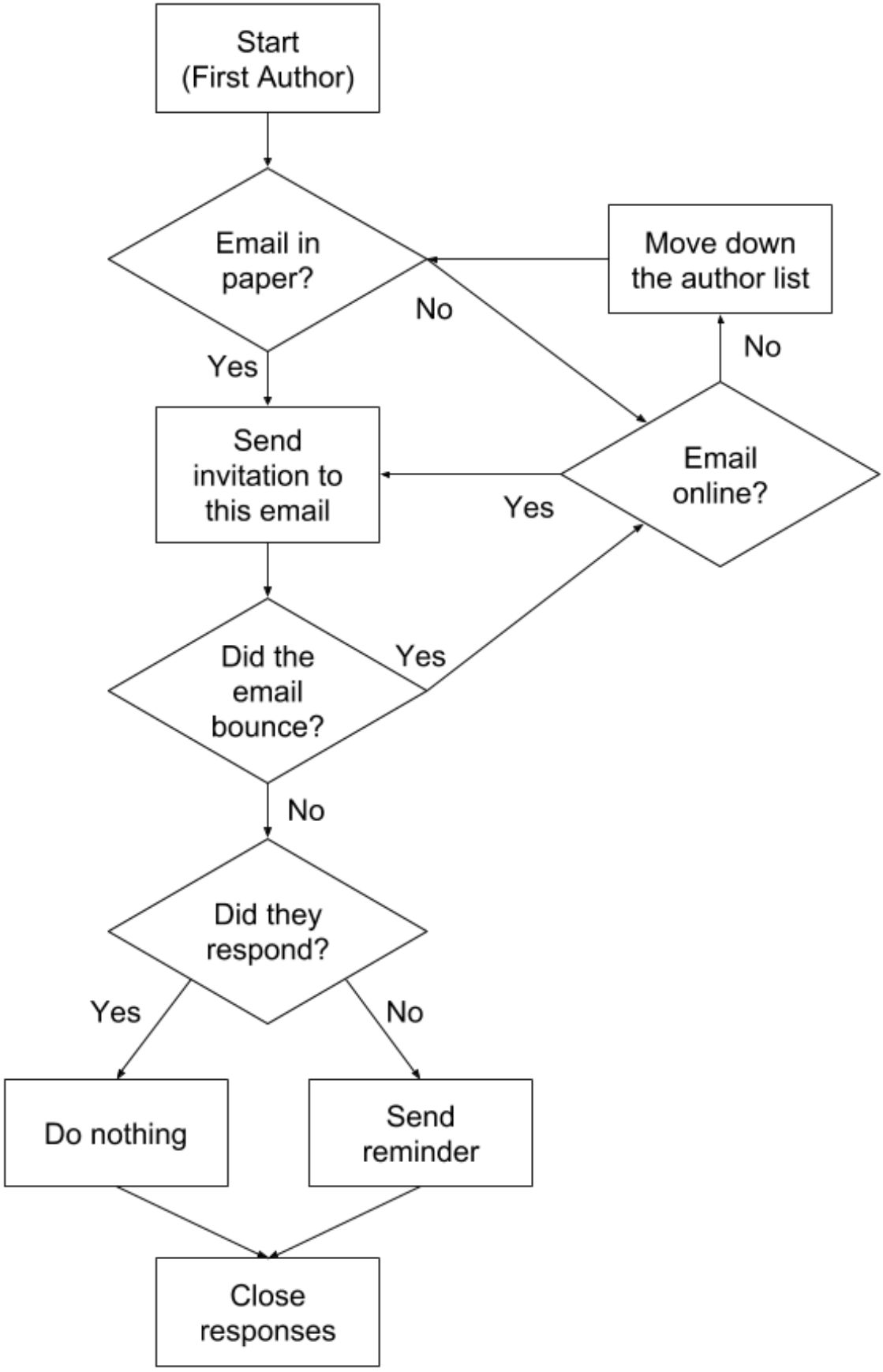


Figure 5.3: The survey dissemination process

determine whether the dissemination of invitations was successful before emailing the larger group of participants.

I received one response to the survey during the week following this invitation and deemed the dissemination successful, then I sent invitations to the remaining first authors. At this point, I noticed a number of emails had "bounced" back to me. For each of these bounced emails, I searched the web for another email address for the author. Often, this occurred because the author moved on to another place of employment or study. I emailed participants again with new addresses where possible. Of the 253 papers, there were 18 first authors that could not be reached. At this point, we turned to the remainder of the authors for these papers. Following the same protocol for first authors, I used email addresses from the papers first, then turned to publicly available email addresses. I sent invitations to 17 second authors and one third author, reaching exactly one author for each of the 253 papers in our sample. It is important to note that not all of these invitations were sent to unique authors; 14 individuals were the first authors on two ICSE papers in the time period in the study. An author responding twice was still relevant to the study, as their responses would differ for each paper. At this point, we had received 44 responses, for a total response rate of 17.39% by paper.

To maximize the response rate without risking multiple responses per paper or spamming participants, I wrote a simple email script for authors who had not yet responded to the survey, reminding them of the survey and communicating the response deadline. These reminder emails led to an additional 16 responses for a total of 60 responses, a response rate of 23.72%. Of these responses, one participant responded to the survey twice for two separate papers. For the purposes of analysis, since their responses were for different papers and were therefore unique, we will consider them as two participants. This response rate is well beyond the typical 10% response rate for surveys, despite being complicated and involving a number of long-answer questions. This may be due to researchers wishing to discuss their work with others, or an intrinsic motivation to further the cause of research that would be beyond typical survey populations. This concluded our data collection phase.

5.3.1 Participant Population

The survey responses came from a diverse group of researchers. Participants identified a broad range of countries where they conducted the research in their papers, shown

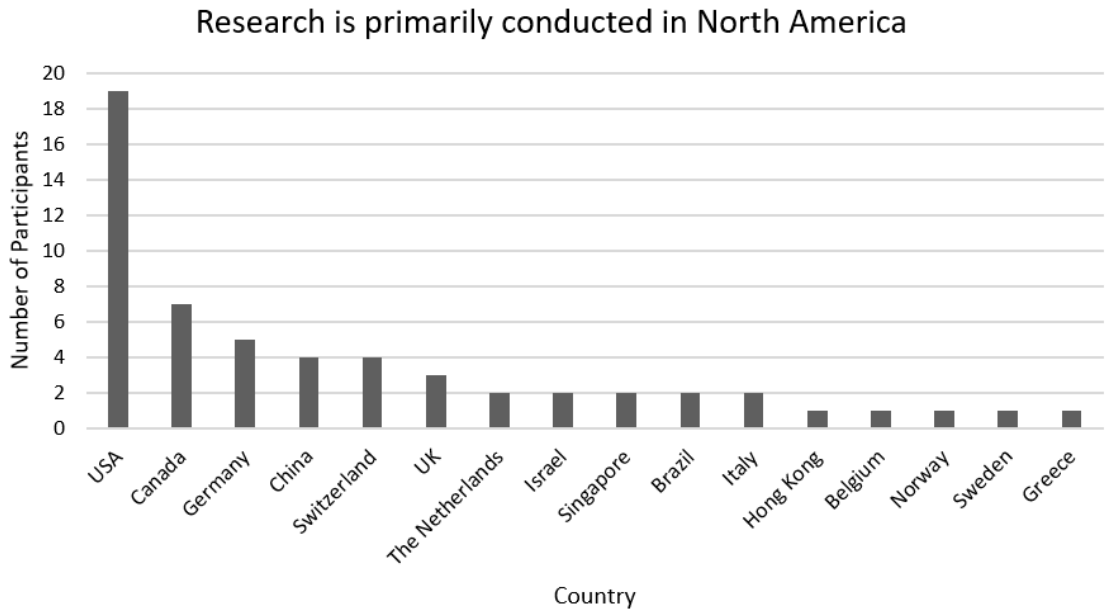


Figure 5.4: Participants indicated the country in which the majority of their research was conducted.

Table 5.1: Participants were split fairly evenly between university faculty and students, with some industry involvement.

Affiliation	# of Participants
University Faculty	28
University Student	26
Industrial Research Lab	2
Student + Faculty	2
Faculty + Industry Developer	1
Faculty + Industrial Research Lab	1

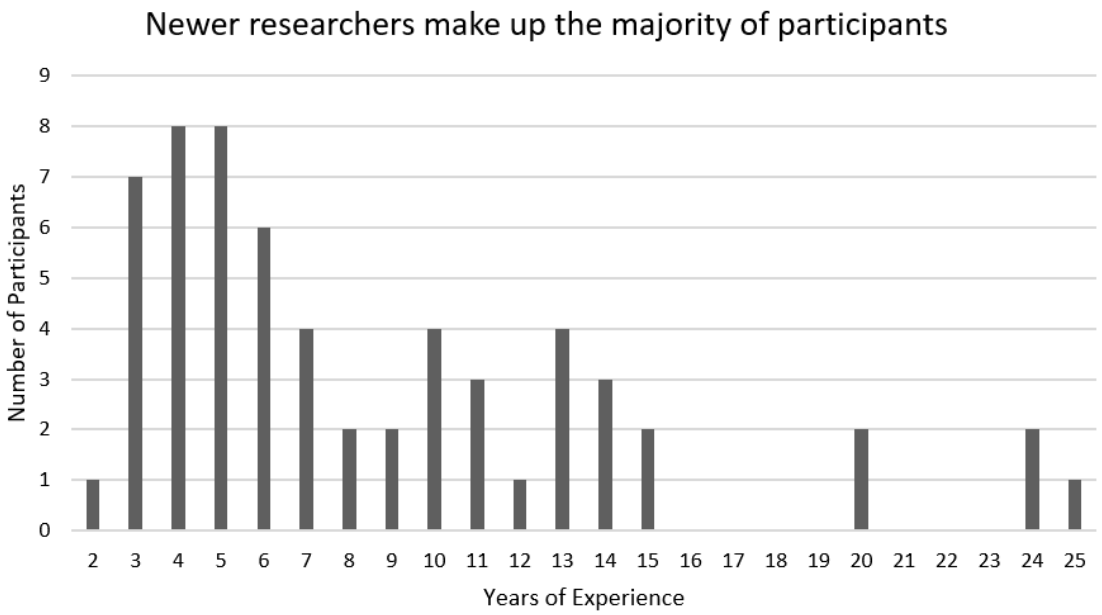


Figure 5.5: Participants had a wide range of experience conducting software engineering research.

in Figure 5.4. There were a large number of participants who indicated that the majority of the research work for their paper was conducted in the United States, and this is what we expected as there are a large number of research institutions in that country. Participants were split fairly evenly between university students and university faculty members with a few industry researchers, as seen in Table 5.1. Participants also indicated that they had a wide range of experience conducting software engineering research, with a minimum of 2 years experience, a maximum of 25 years experience, and a mean of 7.1 years of experience. The distribution of participant experience levels is shown in Figure 5.5. It makes sense that there are a large number of participants with less than 5 years of experience, as Ph.D. and Masters students make up almost half of the participants and they would likely have less experience than a participant who was a faculty member.

5.4 Analysis and Interpretation

The survey addressed all three research questions through both qualitative and quantitative measures. This required using various analysis techniques to capture the intricacies of participant responses and to synthesize quantitative and qualitative analyses into cohesive findings that accurately reflect the data.

5.4.1 Author Classification

The goal of this analysis was to determine, from the perspective of the authors of ICSE papers, what work they conducted and published at ICSE, and compare this information against my own classifications to validate the work conducted in the systematic mapping study.

Information Collection

First, I collected survey responses that were relevant to this analysis task in a spreadsheet. The questions of interest were:

- Your ICSE paper title is: (auto-filled for participant)
- Please select all the research strategies that describe the research conducted in your paper.

- Please elaborate on why you used those strategies.
- Were there any other research methods used as part of the study but NOT reported in the paper? If yes, please elaborate which ones and WHY they were not described in the paper.
- Which data source(s) were used in the research reported in your paper? Please select all that apply.
- Please elaborate on why you chose those data sources.

These responses provided enough information to understand not only how participants classified their papers, but also why they chose those research strategies and data sources for their work to aid investigation into why the researcher and author classifications may have differed. Then I combined these responses with the results of the systematic mapping study, which included my own categorization of the research strategies and data sources for each paper, as well as a small explanation of what in the paper led me to that choice. This spreadsheet view allowed for easy comparison for each paper.

Classification Comparison

Next, I compared my classification with the author's classification of their work without considering any explanations or background information from myself or the participants. For both the research strategies and the data sources, I tagged the paper as one of four types:

- **Full Agreement** My classification was the same as the participant, regardless of how many categories were chosen.
- **Partial Agreement** My classification was the same as the participant for at least one category, but we disagreed about at least one other category. For example, if there was agreement on a Computational Study, but the participant also categorized the paper as Formal Theory where I did not.
- **No Agreement** The participant's classification and my classification were completely different.

- **Other** The participant did not use the classification scheme and rather chose to classify their paper using the "other" category with an explanation. This prevented comparison against my classifications.

After tagging each paper I investigated the causes of disagreement between my previous classification and the participant's classification. For any paper that was tagged as **Partial Agreement**, **No Agreement** or **Other** I read both classifications, both of our explanations, and re-read the paper where necessary. I then wrote a brief description of what I believed caused each disagreement for further analysis.

Card Sorting

After investigating all of the causes of disagreement, I printed each of the descriptions onto paper and card-sorted the probable causes. This led to a series of distinct disagreement types. They were:

- **Lens Miscommunication** It appeared that there was a miscommunication of the research lens and how to classify work correctly. For example, if a participant classified their work as Experimental Simulation while simultaneously explaining that they used entirely computational methods.
- **Researcher Oversight** I missed some details in the paper and classified it incorrectly.
- **Author Oversight** The author did not mention details of their paper that clearly indicate an additional research strategy or data source was used.
- **Formal Theory Noise** This type of disagreement was for research strategies only. For the purposes of reducing noise, my classification did not include minor literature reviews (such as related work or background sections), the proposal of a model as the result of an empirical study, or the description of an approach, model, or tool as a Formal Theory study of its own. I considered this just a part of the context of the other work that formed the bulk of the paper. This was not communicated in the survey, so some participants categorized these parts of their papers as Formal Theory.
- **Formal/Theoretical Noise** Similar to Formal Theory Noise, this type of disagreement is for data sources only. The Formal/Theoretical data source category was meant to refer only to *entire papers* that did not include any of the

other data source categories whatsoever. This appeared to be unclear, as some participants used this to refer to a study that formed *part* of their paper that did not include any data sources (even though there was another part of their paper that did utilize one of the data sources). Some participants also appeared to use this categorization because of the formal or theoretical nature of their topic, even though they acknowledged using one of the data sources.

- **Fringe Case** The participant and I both had a good case for why their paper should be classified a certain way because the model doesn't clearly capture their particular work.
- **Unclear** It was not apparent why the participant categorized their work this way, as they did not leave enough information to determine the cause of the disagreement.
- **Descriptive Agreement** The participant selected "other" (or one of the provided options) when categorizing their work and explained their work. In this case, the participant's *description* of their work aligned with my description of their work, so there was an agreement of a descriptive nature.
- **Information Missing** The participant referred to details of their work that I could not find, such as an interview study for which there were no details in the paper. In this case, I assumed that they conducted this work as a part of their overarching research process, and published this information elsewhere or not at all.

Because some papers included the use of many different data sources and research strategies, some papers had more than one category of disagreement. To investigate the reliability of these assumptions I conducted member checking with participants, which I describe in Section 5.5.

5.4.2 Long Answer Question Analysis

This goal of this analysis was to develop findings from the long answer questions in the survey. During this phase, I recorded diaries that explain the changes that I made each time I worked on the analysis and why I made the changes so that it would be possible to trace my thought patterns from initial codes to synthesized findings. These diaries are included in Appendix C.

Collecting Participant Responses

There were two separate analyses conducted with identical methodological approaches. The first was to answer RQ2. This included the same questions and responses as the analysis described in Section 5.4.1. However, the previous analysis was focused on determining whether the systematic mapping study classification could be considered valid, while this analysis was focused on understanding why participants chose the research strategies and data sources they reported in their ICSE paper.

The second long answer analysis was primarily concerned with addressing RQ3. It included responses to the following questions:

- How do you define triangulation, and how do you use triangulation in your work?
- In your own work, do you PRIORITIZE any criteria (generalizability, control, or realism) over another? Why? Please elaborate.
- Do you PERCEIVE A BIAS in the software engineering research community regarding certain criteria (generalizability, control, or realism) when it comes to publishing work? Please explain.
- Would you like to comment or add anything else?

These analyses were separated for two reasons. First, one analysis was focused on the participant's ICSE paper, and the other was focused on their research career as a whole, so the context surrounding the responses was different. Second, responses about ICSE papers contained a lot of sensitive and identifying information, whereas the general nature of the questions about RQ3 meant that it would be possible to anonymize some analysis documents for traceability.

The responses to the questions for each analysis were collected in a spreadsheet from the raw survey responses, and from there a text file was created for each participant and entered into an RQDA project [14].

Open Coding

After collecting the information and setting up an RQDA file I performed open coding on the responses. In this way, I could analyze the information without considering any particular hypothesis or theoretical underpinning to the coding, allowing for the

findings to be grounded in the data. The first round of coding involved adding as many codes as necessary to capture all of the information offered by the participants. An example of this large code set is shown in Figure 5.6.

The second round of coding involved reading and understanding the responses that went into each code and condensing the code set. This included merging redundant codes, renaming codes, and deleting codes that provided little value to the analysis until each code provided unique and valuable insights for synthesis into cohesive findings. For example, I merged the code "Not using triangulation" into the code "Triangulation practices". This is because when a participant described why they did not use triangulation techniques it allowed me to show the contrast between the participants who described extensive triangulation practices and those who chose not to triangulate. For the RQ3 analysis, as there were codes that pertained to distinct questions and topics, there was a third round of coding to group these codes into categories. For example, one of the code categories was **Community Priorities**, which contained the following codes (where "SERC" is an acronym for "Software Engineering Research Community"):

- Control not prioritized in SERC
- Control prioritized in SERC
- Generalizability not prioritized in SERC
- Generalizability prioritized in SERC
- Realism not prioritized in SERC
- Realism prioritized in SERC
- No bias in the community

For RQ3, another member of the research team independently coded a subset of the data as a form of coding validation. The researcher was provided with the subset of data and a coding instruction package that explained each code. This coding validation package is included in Appendix D. After she independently coded this data, I compared her codes against my own. After identifying some minor disagreements, we discussed them and reached an agreement. There were very few cases of disagreement, and any found were primarily errors of omission by my colleague due

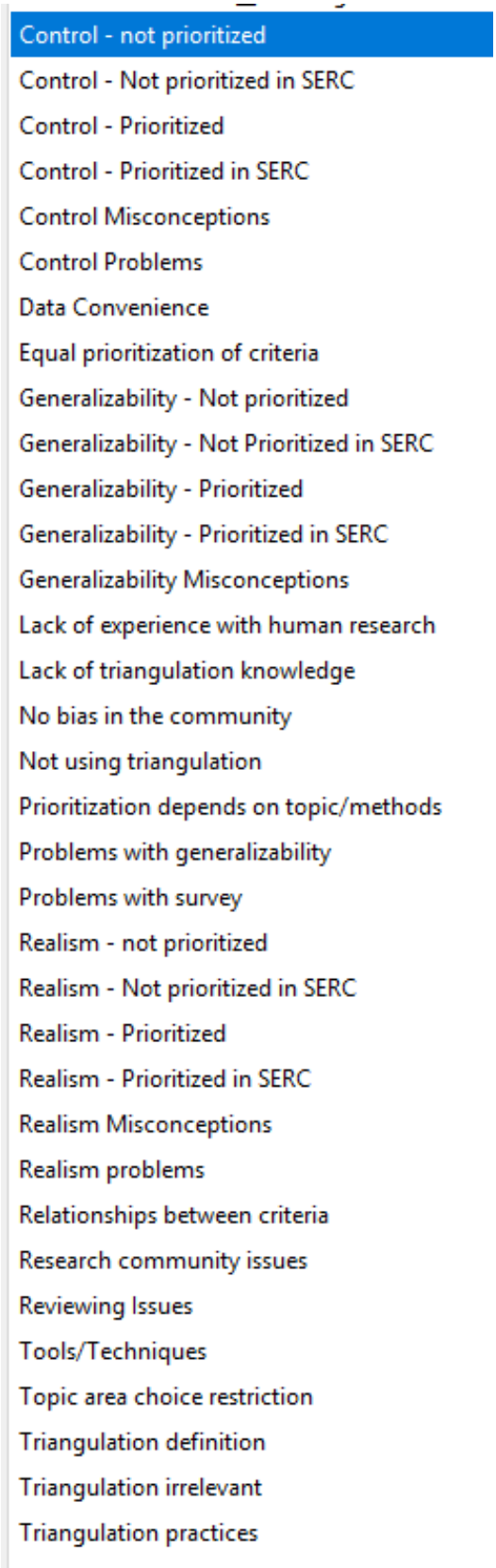


Figure 5.6: RQ3 analysis codes after round one.

to the sheer volume of codes in the code set, or minor disagreements where participants used wording that was vague. I did not conduct a similar validation step with the RQ2 analysis, as it was far simpler than RQ3 and would likely produce even less disagreement than the previous exercise.

Synthesis

Having completed the coding for each analysis, it was necessary to bridge the codes and synthesize the findings. I read the text that informed each code and wrote a description of the information, recording the quotes that were most informative to my description. Following the exercise of describing the codes, I wrote a brief summary that explained the content of each code in one to two sentences. After describing and summarizing all of the codes in a category, I collected the summaries so I could see the connections between the codes. I used highlighters to connect codes that were related to each other, shown in Figure 5.7. After this process, I synthesized the codes into a descriptive text that explained the information included in the codes in a cohesive way that connected related information.

I conducted this process for the codes in RQ2 (which formed a single category), as well as the code categories of RQ3. One code category, Systemic Issues, was not synthesized because much of the data was already captured through the other code categories, and it contained a number of unique assertions of issues within the software engineering research community that were not shared by other participants. Anonymized records from this process is included in Appendix E, where I present the analysis and interpretation document for the code category "Reasons".

5.4.3 Statistical Analyses

Since the systematic mapping study and the survey contained categorical data, we also conducted some simple statistical analyses to better understand and present this data in the form of visualizations. We used R [31] to visualize the data, and the resulting figures are presented in Chapter 6.

To present the data from the systematic mapping study, we visualized the research strategies and data sources separately in simple bar graphs showing the percentage of papers that included each of the categories. Then we visualized the "balance" of the desirable research criteria by showing the percentages of each research strategy in their respective segment of the circumplex.

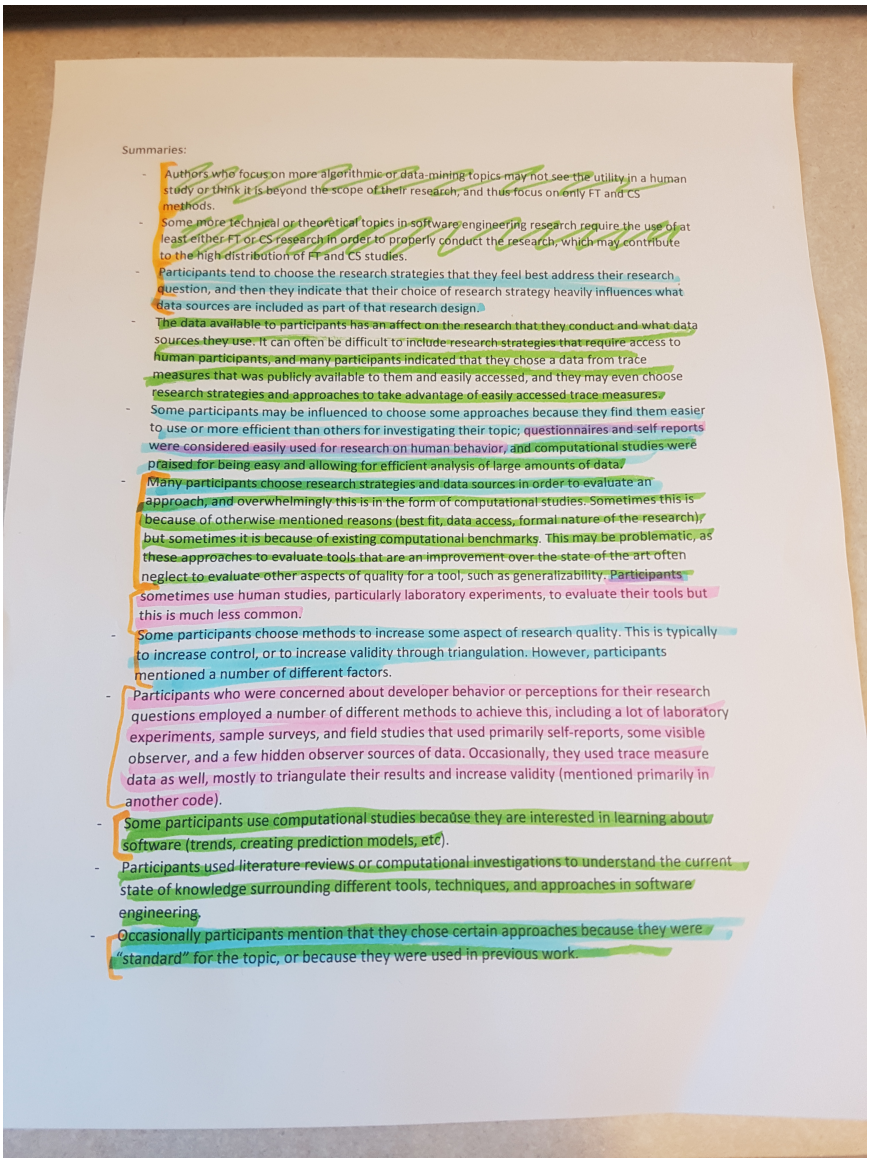


Figure 5.7: The process of connecting related codes.

There were two types of visualizations for the survey data. First, for the participant demographic data, I created a series of simple graphs to demonstrate the diversity of my participant population (shown in Section 5.3). Second, there were responses to Likert-scale questions, which we visualized using R Likert charts.

5.5 Member Checking

Following the analysis, I conducted member checking for the findings of the systematic mapping study validation. My goal with this task was to see whether or not I could reach an agreement with the participants whose paper classifications differed from mine. This was not an easy process, as it was difficult to approach the participants in such a way that did not influence their responses or rely on the use of our models, because that could cause further miscommunication between myself and the participant. For each participant in the study, if there was some form of disagreement and they indicated that we could contact them to learn more about their responses, I sent them an email asking for them to clarify my understanding of their work.

In the email, I briefly thanked them for their participation in the survey and said that I wanted to clarify their responses, and reminded them of their ICSE paper title. Then I used the language that the participant used in the survey as well as their paper to describe what they indicated in the survey about their paper, my understanding of their work, and asked them if I had accurately described their paper. The form letter is provided in Appendix B. For participants for which there was a disagreement but the participant did not consent to follow-up questions, I moved forward with my classification for all categories other than **Researcher Oversight** and **Fringe Case**, where I selected the participant's classification.

I sent a total of 32 member checking emails (one participant was sent member checking information about two of their papers). After receiving replies from participants I discussed them with my collaborators to determine whether or not the participant and I had reached an agreement about the work reported in their paper. We used this information to validate my assessment of the results of the paper classification task completed by the participants, and updated our inter-rater agreement statistics and final systematic mapping study classifications accordingly.

Chapter 6

Findings

In this chapter, I present the findings from the systematic mapping study and the survey. I outline the findings of each analysis task described in Section 5.4 individually, as each analysis task resulted in unique and valuable insights. Collectively, this chapter answers not only the three research questions but also other questions that we addressed with our methodology to support our findings. To protect author (survey participant) anonymity, author quotes are identified in this section using an identifier, "Ax", where the x corresponds to the order in which authors responded to the survey.

6.1 RQ1: What research strategies and data sources are described in research published at ICSE?

To answer this research question, first I present the initial findings from the systematic mapping study. Then I discuss the results of the author classification task from the survey, which indicated the need to make some corrections to the classification data. Finally, I present the findings from the systematic mapping study after having applied those corrections.

6.1.1 Initial Findings

After classifying each ICSE paper according to which research strategies and data sources were reported, we collected that data and plotted it in bar graphs, shown in Figures 6.2 and 6.1.

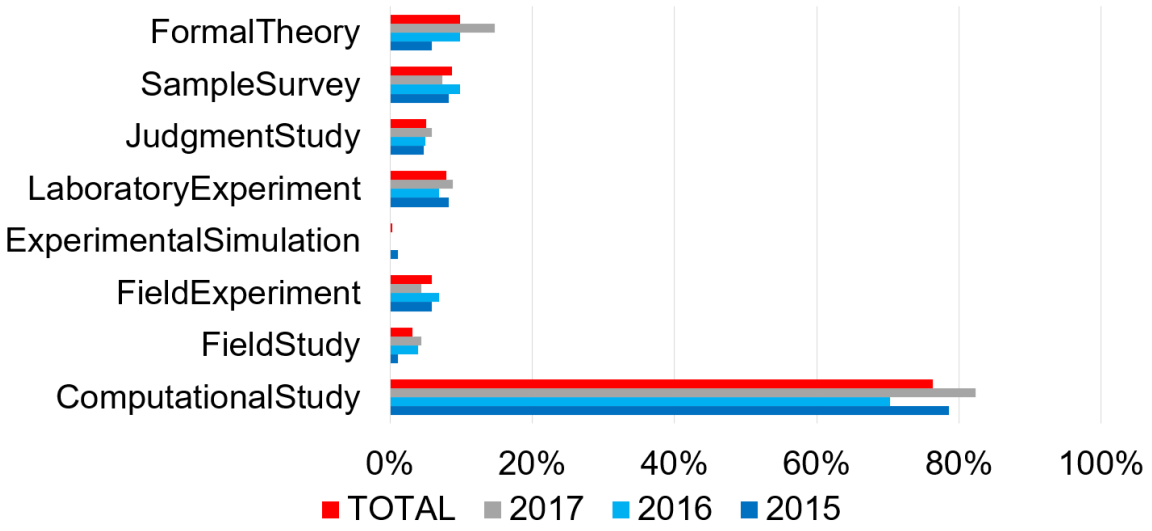


Figure 6.1: Research strategies included in ICSE papers before systematic mapping study validation.

Here we see an indication that Computational Studies and Trace Measures are prominent at ICSE, and that there are very few instances of Field Studies and Archival Records data. I elaborate further on the findings in Section 6.1.3 after I applied the necessary corrections from the systematic mapping study validation.

6.1.2 Systematic Mapping Study Validation

To validate the study, we asked authors in the survey to classify their own papers so we could calculate a measure of inter-rater reliability from the classifications, as described in Section 5.4.1. The results of the surface-level comparison of my classifications of ICSE papers versus the classifications given by the paper authors are shown below in Table 6.1.

Table 6.1: Inter-rater agreement between myself and ICSE paper authors.

Type	**Research Strategies**	**Data Sources**
Full Agreement	15	18
Partial Agreement	31	20
No Agreement	9	16
Other	5	6

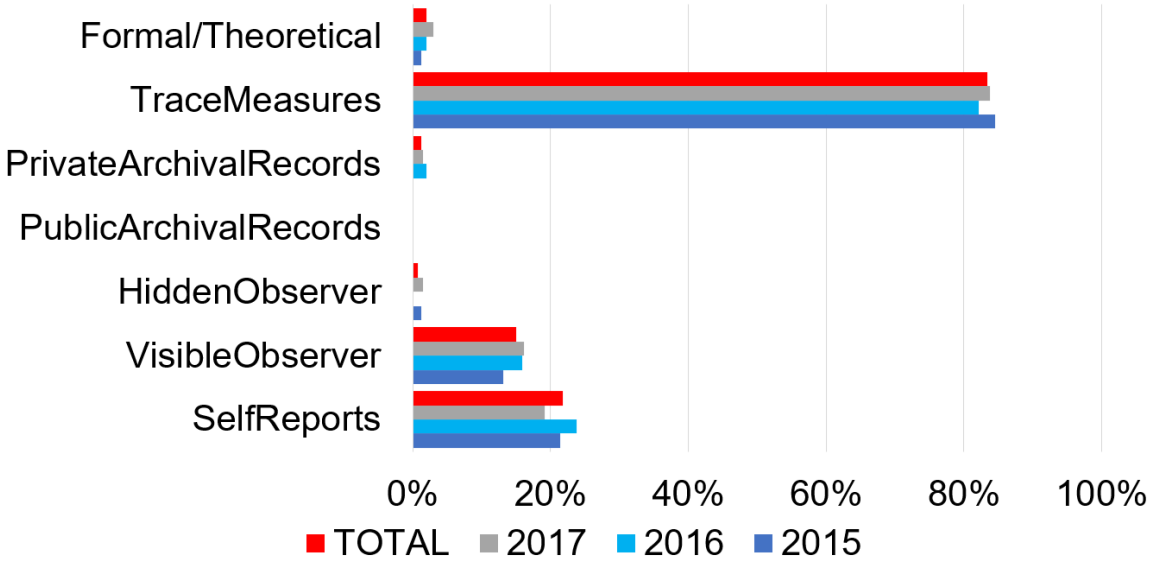


Figure 6.2: Data sources included in ICSE papers before systematic mapping study validation.

If we consider the cases of Full Agreement we see an inter-rater reliability level of 25% for research strategies and 30% for data sources, which is well below what is generally considered acceptable. However, when we closely consider possible causes for the disagreements the results are much more positive, as shown in Table 6.2. Some papers had two types of disagreement; in these cases, there was more than one disagreement type for a single paper, each with differing probable causes.

Here we see that the majority of disagreement appears to have been caused by issues of miscommunication and noise rather than true disagreement about how to classify the papers or because of incorrect researcher classification.

To calculate a new measure of inter-rater reliability that takes into consideration the reasons for disagreement, we considered all responses to be in agreement with the systematic mapping study unless they included disagreements because of Researcher Oversight, a Fringe Case, or a case where the disagreements cause was Unclear due to a lack of information. This led to an 80% agreement in the classification of research strategies in papers, and an 83.3% agreement in the classification of data sources.

Following this step, I conducted member checking with authors where possible to further investigate these disagreements, as described in Section 5.5. Of the 32 authors contacted I received 27 responses, which were overwhelmingly positive in terms of agreement with my description of the authors' work. There were a few cases where the authors disagreed with my description of their work, but this had to do mostly

Table 6.2: Causes of disagreement in categorization.

Cause of Disagreement	**Research Strategies**	**Data Sources**
Lens Miscommunication	12	10
+ Researcher Oversight	2	1
+ Author Oversight	0	2
Researcher Oversight + Author Oversight	5	2
Author Oversight	5	7
+ Formal Theory Noise	2	0
+ Fringe Case	0	1
Formal Theory Noise	11	0
Formal/Theoretical Noise	0	3
+ Information Missing	0	1
Fringe Case	2	4
Unclear	2	2
Descriptive Agreement	3	6
Information Missing	0	3

with incorrect wording and issues of oversimplification on my part. This validated my assumption that most of the disagreements were not truly "disagreements" at all but rather caused by issues of miscommunication and unclear instructions. However, there were still some cases where papers were clearly classified incorrectly, and we made changes to the systematic mapping study classifications to reflect this. I changed my classifications where there was a case of Researcher Oversight or a Fringe Case, but I kept my initial classifications in all other cases. In total, I corrected the research strategy classifications for ten papers and the data source classifications for eight papers. The findings presented in the remainder of this chapter are based upon having made these corrections.

6.1.3 Corrected Findings

The corrected classification data is shown in Figures 6.3 and 6.4. I also show the differences between the initial findings and the corrected findings in Tables 6.3 and 6.4.

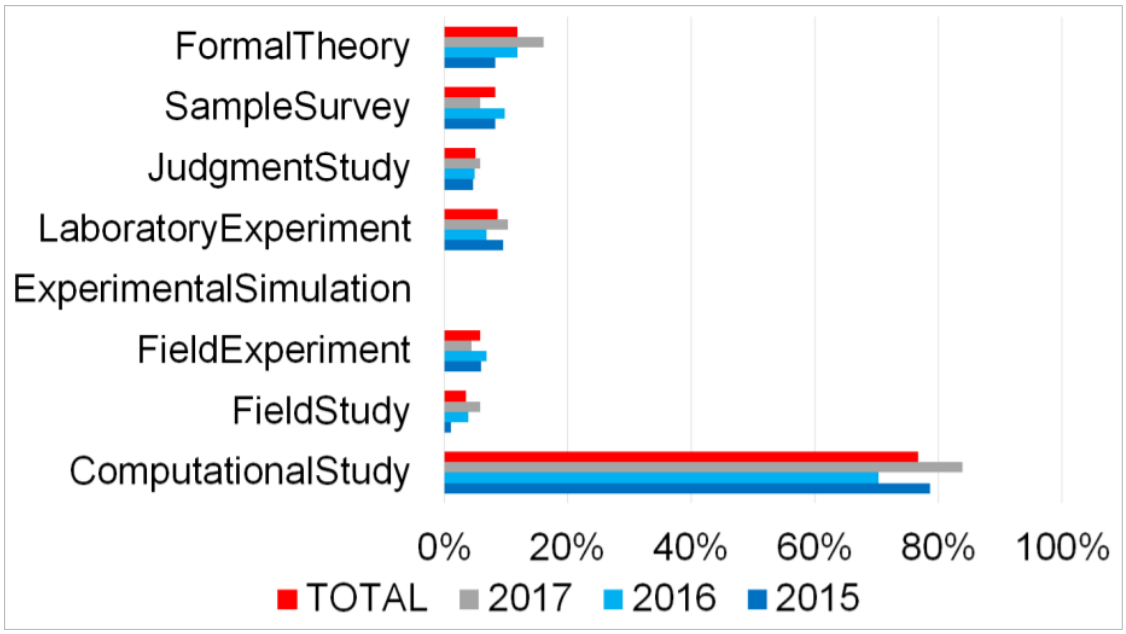


Figure 6.3: Research strategies included in ICSE papers, corrected after member checking.

Here we see a high use of Computational Studies (76.68%), and a high use of Trace Measures as a data source (82.21%). This reflects the percentage of papers that include that research strategy or data source; papers reported up to three research strategies, and up to four different data sources. We found that 48 papers reported more than one research strategy, and 53 papers reported more than one data source. We also found no instances of the use of an Experimental Simulation in the sample; this was not unexpected, as replicating a specific natural environment for a participant in a software engineering experiment is uncommon.

Computational Study was the most commonly used strategy, but the papers reported on a wide variety of different types of studies that used this approach. Computational Study included data mining studies, natural language processing experiments, computer simulations, computational experiments to evaluate tools and techniques, computational analysis of software artifacts, and computational prediction models, to name a few. The Trace Measures used in papers also varied significantly, with researchers reporting the use of log files, data from websites such as StackOverflow, datasets of software bugs, open source repositories, code comments, research papers, and software programs such as websites and apps, among other sources.

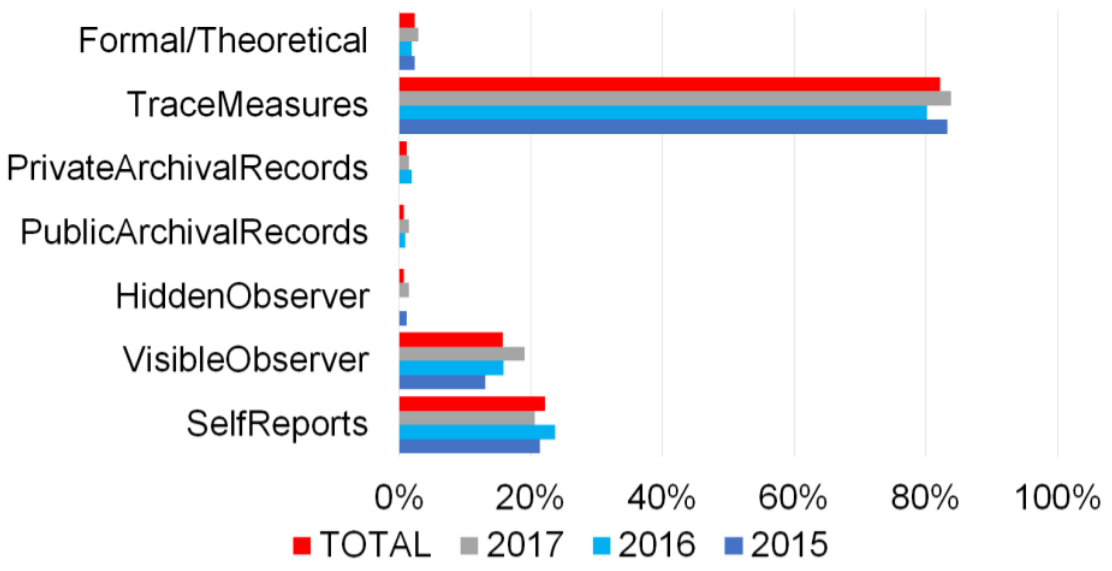


Figure 6.4: Data sources included in ICSE papers, corrected after member checking.

Table 6.3: Summary of corrections to research strategy classifications.

Research Strategy	**Initial**	**Corrected**	**Difference**
Computational Study	193	194	+1
Field Study	8	9	+1
Field Experiment	15	15	0
Experimental Simulation	1	0	-1
Laboratory Experiment	20	22	+2
Judgment Study	13	13	0
Sample Survey	22	21	-1
Formal Theory	25	30	+5

6.2 RQ2: Why do authors choose the research strategies and data sources they describe in their papers?

Authors identified a variety of reasons for choosing data sources and research strategies for their work, describing everything from general approaches that apply to all of their studies, to specific factors that explained the design for the research described in their ICSE paper. Often these reasons were intertwined, and authors mentioned tradeoffs in balancing a variety of factors.

Table 6.4: Summary of corrections to data source classifications.

Data Source	Initial	Corrected	Difference
Self Reports	55	56	+1
Visible Observer	38	40	+2
Hidden Observer	2	2	0
Public Archival Records	0	2	+2
Private Archival Records	3	3	0
Trace Measures	211	208	-3
Formal/Theoretical	5	6	+1

The most important factor is choosing the best research design to address your questions. The biggest theme that we saw in this analysis was that authors tried to choose the research strategies that were the best fit for their research question or topic, choosing the strategies that were "most suitable to answer our rq" (A43) or "appropriate for this type of research" (A53).

Authors who were focused on formal or algorithmic methods indicated that primarily Formal Theory and some Computational Studies were the best approaches for their work. Likewise, authors who focused on technical research or were interested in learning about software itself indicated that they used primarily Computational Studies in their work, with some Formal Theory. Mostly, authors indicated that they used these strategies to evaluate an approach, tool, or algorithm that formed part of their research contribution. This makes sense because software engineering research is often solution-focused and authors need to demonstrate the quality of their solutions to be accepted for publication. For example, authors said that they conducted Computational Studies "to provide real-world evidence to the motivation for our work as well as prove [the] scalability of our approach" (A10) and to "[test] performance against previous benchmark suites" (A45).

Other authors said they selected human-involved research strategies like Laboratory Experiments, Field Studies, and Sample Surveys because they were the best choice for addressing their questions. Their research interests varied but tended to center around Field Studies and Sample Surveys when the goal was to understand some aspect of developer perception or behaviour, and leaned towards more Laboratory Experiments when aiming to understand the effects of some tool or approach on developer behaviour and performance. For example, one author stated that for their research questions "the best strategy was to enter the company and investigate the experience of professionals with the practice under study" (A31).

Researchers want to achieve aspects of research quality through their choices of research strategies and data sources.

One factor that authors mentioned was that they wanted to achieve some aspect of research quality with their design. Sometimes this was one of the three criteria mentioned in our study, with one author who prioritized control saying that they chose research strategies and data sources to "isolate the difference between two groups with most variables being as controlled as possible" (A15). There were also a number of research quality goals beyond these criteria; one author said they wanted to "have a large, varied collection of data, representative of a variety of programming styles" (A41), while another author was influenced to design their study in such a way that results would be "grounded in empirical data" (A4).

Data source needs to match the research strategy.

The choice of a data source was also closely linked to the choice of research strategy, with many authors indicating that they chose their data source because it was the best fit for the research strategy they selected. For Computational Studies, authors said they used Trace Measures (particularly code repositories and benchmark datasets), where authors who conducted human-involved studies indicated using primarily Self Reports and some Visible Observer data as "a result of the study design" (A3). This makes sense because some research strategies traditionally rely on a specific type of data, like observations in Experimental Strategies and Self Reports for Respondent Strategies.

"User studies would have been beyond the scope of our work" (A27)

Some authors who were focused on technical research topics also indicated that they thought that the other strategies, ones involving active human participation, were beyond the scope of their work and they did not see them as an option, saying that "no human participants or empirical evaluation were necessary" (A56) and "because we are working on algorithms [...] there is no need to conduct user study with human participants" (A21).

As mentioned above, most of these authors used computational approaches to evaluate approaches or tools. Some authors *did* indicate that they included humans to evaluate their tools, though this was less common. For example, one author said that they "used a controlled environment for participants to use a tool we developed"

(A59), and another said they used "Self-reports to assess perceived usefulness" (A18) of a tool. We suggest not only that human-involved studies *are* within the scope of most of these types of papers, but that the perception that humans cannot be included in more technical research areas could be problematic. One author pointed out that "there is too little regard for generalizability when attempting to determine whether technique a is better than technique b" (A47), suggesting that we should consider including more human evaluations of tools to determine how well they generalize to different populations of developers.

Authors are limited by participant access barriers and tempted by easily collected data.

Many authors said that the availability of certain types of data influenced their research design. Authors said it was difficult to gain access to developers and software engineers to conduct human involved studies, and even with access to human authors, it was very difficult to use some types of data, such as Hidden Observer data, because of ethical concerns. One author even said that "ideally, we also would have conducted a field experiment to answer questions regarding usability, but we didn't have subjects readily available with the right training" (A16). This suggests that researchers could be discouraged from triangulating with human-involved research strategies because of a lack of access to an appropriate population.

On the other hand, authors found it much easier to gain access to Trace Measure data, saying it was "publicly available" (A34, 54, 17), easy to access (A53), or used by other researchers in related work (A34). Although a few authors said that Sample Surveys are an easy way to reach practitioners (A10), other authors explained that they chose Computational Studies with Trace Measures because they were easier to conduct and more efficient at analyzing large sample sizes than other research strategies (A41, 57).

While certain pragmatic choices must be made when designing a study, choosing research strategies and data sources to take advantage of approaches that may be more convenient and publicly available data may be problematic. As one author pointed out:

> "I see construct validity issues in much of SE research. Just because data is convenient or available does not mean it reveals what we are looking for." (A4)

6.2.1 Summary

Overall, this shows that authors included many factors when deciding on the research strategies they employed and the data sources they used to conduct their research. While most authors indicated that their selection was the best fit and most common approach for their research questions, there were some problematic reasons for choosing a research design that arose from the survey. Some authors thought that human-involved studies were beyond the scope of their research, or chose Computational Studies with Trace Measures because the data was more convenient to access or because it was more efficient than other approaches. This, combined with the majority of authors indicating that their research area was technical, theoretical, or aimed at understanding software as it exists, gave us insights on why Computational Study is the most commonly used research strategy in ICSE papers.

6.3 Are ICSE papers a good representation of the overall research careers of its authors?

We recognized that a single ICSE paper might not be representative of a researcher's entire body of work and that they might send certain types of work to different venues. Therefore, we asked authors how often they used each of the eight research strategies in their last five years of research according to a Likert scale. We did not specify a publication venue, or that the research even resulted in a publication. The results from this question are shown below in Figure 6.5.

Once again, we see that Computational Study is the dominant research strategy. Authors indicated that Formal Theory, Laboratory Experiments, and Sample Surveys were next on the list and that the other research strategies were much less commonly used. The only major discrepancy between the results of the systematic mapping study and this analysis is Experimental Simulation; I did not encounter a single case of this approach in the sample, yet a number of authors indicated that they used this strategy. The potential explanation for this discrepancy is that there was a miscommunication of the research lens, and authors were using Experimental Simulation to refer to computer simulations, which was indicated by the author paper classification in Section 5.4.1.

This supports our systematic mapping study, showing that the breakdown of research strategies used at ICSE may be an accurate reflection of the overall research conducted by ICSE technical track authors.

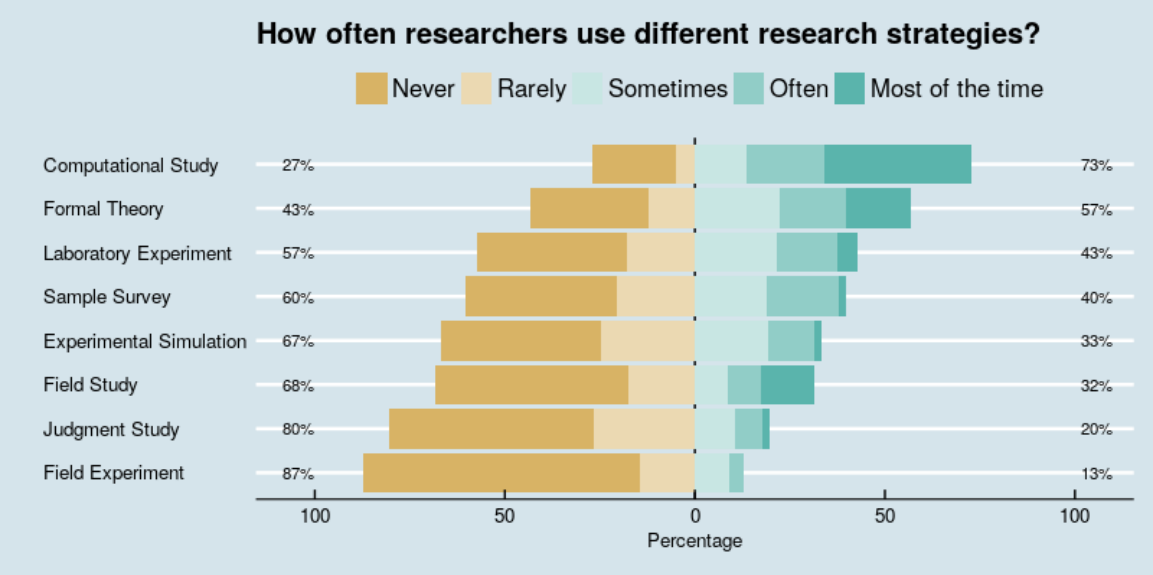


Figure 6.5: Authors indicate a high use of Computational Studies in their overall research careers, regardless of publication venue.

6.4 What about triangulation?

There is a wide variation in our understanding of triangulation as a community. For context, I understand triangulation as *"taking different angles towards the studied object and thus providing a broader picture"* of the phenomenon under study, as described by Runeson and Höst [32]. Some authors provided us with rich definitions of triangulation and indicated that they utilized a number of different techniques in their work. Two examples of such explanations given by authors are:

> "Triangulation means integrating different sources of evidence. You can triangulate across data collected from different sites, using different methods, or analyzed in different ways. I do all of the above, depending on the study." (A12)

> "Triangulation can be done: across researchers, across data sources, across participants, across methods. Data triangulation is to ensure that whatever data you analyse actually corresponds to other data sources – does

> the evidence point in the same direction? In a general sense: any form of triangulation aims to find corresponding evidence through different means (researchers, analysis, methods, etc.) so as to ensure reliability of a study's findings." (A46)

Some authors were not as diverse or general in their explanations. Some researchers only described their knowledge and practice of triangulation in the context of their own specific research domain, particularly in the realms of experimentation and tool evaluation. For example, an author whose work was experimental in nature said:

> "I define triangulation as reaching conclusions based on multiple data sources and/or multiple experiments that investigate some phenomenon using different techniques. Often.. I try to use multiple data sources when applicable." (A60)

This does not necessarily indicate that these authors do not understand triangulation in a more general context, but they may only apply it in a certain way in their work.

Surprisingly, a large portion of authors did not understand triangulation in the context of research. Many even indicated that they had never seen the word before. Some examples that we found were:

> "I've never used this word; I'm unsure what is being asked exactly (perhaps that answers something?)" (A16)

> "I did not know about triangulation prior to this survey." (A56)

This broad spectrum of triangulation knowledge and understanding within our community suggests the need for education surrounding the importance of triangulation, which would facilitate the use of more diverse research strategies and data sources, compensating for the limitations of currently used techniques and increasing the quality of our collective body of work.

6.5 RQ3: What is the "balance" of quality aimed for across generalizability, control, and realism in research published at ICSE?

To investigate a potential imbalance balance of generalizability, realism, and control in ICSE papers we populated the segments of the circumplex according to the percentage of papers that included each respective research strategy, shown in Figure 6.6. Due to the high use of Computational Studies, our visualization shows a skew towards relatively high potential for realism and generalizability, but low potential for control. Neither criteria's potential is fully maximized like it would be if we observed a high number of Field Studies or Sample Surveys/Formal Theory. However, this visualization just shows the potential for each of the three criteria, it cannot show us whether or not the researchers actually maximized these criteria in their work.

In order to investigate the notion that levels of realism and generalizability may be higher than control in ICSE papers, we asked authors in the survey to rate the levels of generalizability, realism, and control in their paper. The resulting distribution is shown in Figure 6.7. Here, we see that authors rated their own papers more highly on realism and generalizability and lower on control. Overall, authors rated their papers highly on all three criteria; this was expected, as the papers were published in a top-tier conference and are likely to be of good quality, and authors are unlikely to respond in a way that reflects poorly upon themselves. However, it is the difference between each of the criteria that we would like to draw attention to, which supports the balance indicated by mapping the research strategies to the circumplex.

6.6 What is the balance of our priorities as individual researchers?

Authorss mentioned a number of factors that influence their prioritization of generalizability, realism, and control in their work.

Authors think that generalizability, realism, and control are all equally important.

Many authors said that they highly valued all three criteria equally, but recognize that certain trade-offs exist that make it difficult to prioritize generalizability, realism, and control simultaneously. Those that valued the criteria equally discussed

the different ways that you should prioritize each criterion depending on the research strategy that you choose, which was in alignment with their position on the circumplex. They also explained how they used triangulation with different research strategies to achieve maximization of all three criteria. One author explained that "[prioritization] depends on the purpose and nature of the study. Control is the point of a lab experiment. Realism is the point of a case study. Generalizability is (sometimes) the point of a questionnaire." (A12) This is important because it illuminates the fact that authors understand these concepts without being presented with the circumplex to explain the relationships between the research strategies and the criteria.

Authors are concerned with achieving other research quality goals.

Some authors were more concerned with prioritizing some desirable criteria that they value outside of our research lens. For example, one author stated that they prioritized "rigour, reproducibility, replication and relevance" (A35) in their research rather than the three criteria we studied. Another said they do not prioritize, but that their work "[follows] the empiricist tradition and approach" (A55).

Control is the priority in experimental work.

Those who highly prioritized control did so because they focused primarily on experimental work, because "if you cannot control the variables, the experiment result is meaningless." (A39) Other reasons for prioritizing control were because of increased reliability, applicability, or accuracy, with authors stating "I prioritize C because it makes the results of my experiments more reliable" (A42) or "without control, it is not clear where [the research] is applied" (A32).

Long term impact and applicability calls for generalizability.

Those who prioritized generalizability were primarily concerned with long-term impact and applicability of their tool to populations outside of their own study. For example, one author said that it is "important to make sure that the approach could generalize for subjects beyond the current study to ensure the applicability of the approach" (A22). Authors were also concerned with generalizability because of publication, saying that "if an approach is not generalizable, it makes no sense to others" (A14) and "that's what reviewers easily criticize" (A19).

Realism needs to be maximized to solve industry problems.

Those who prioritized realism tended to have relationships with industry and were concerned with creating solutions to real-world problems that could be adopted by their industrial collaborators as well as other developers with similar issues. For example, one author said "realism comes first. Given that I work [in] an industrial research lab and all my research needs to help practitioners in the real-world daily work" (A10). Another explained that "I need to have impact on the process and save money for the company. Not being realistic means I will have zero impact" (A31).

There are perceived relationships between the criteria.

Another reason why authors may have prioritized generalizability and realism over control is the perception that realism is related to generalizability. Authors believed that realism was necessary to judge generalizability, and that control had to be judged separately. For example, one author stated that they believe "realism is an important contributor to generalizability; the two are not orthogonal" (A9), with another stating "control is often in opposition to the other two, necessitating different types of studies" (A28). This perception of relationships between the criteria points at a lack of community consensus and shared understanding surrounding the three research criteria in software engineering, as these criteria were not linked this way in Runkel and McGrath's original circumplex model [33].

First realism, then generalizability, then control comes last.

Additionally, many authors prioritized two criteria together over another. Overwhelmingly, authors were most concerned with realism first, and then either included generalizability out of a desire to create an approach that could be adopted outside of the developers they studied, or they occasionally included control to ensure that their experimental results would reflect the reality of software development. This aligns with the results of our systematic mapping study, where the circumplex was imbalanced towards a prioritization of realism and generalizability.

6.7 What do we perceive as the priority of the research community as a whole?

There were some authors who did not perceive a particular bias in the research community towards any specific criteria, but most perceived some form of community bias towards one criteria or another.

There may be a historic bias in the community towards control.

A few authors saw a bias towards control in the community in the form of highly controlled laboratory experiments, primarily using student developers that may potentially lead to invalid results (A9). This is also suggested by the literature, where Salman, Misirli, and Juristo found that student developers perform differently compared to professionals, particularly when they need to follow a development approach that is new to them. [34]

One author saw a problem with the prioritization of control, saying that "there often seems to be a bias towards control and experimental rigor, which sometimes makes it harder to build less controlled and naturalistic studies" (A28), suggesting that this bias may make it more difficult to publish Field Strategy work. One author pointed out that this issue may be in the past, saying:

> "The former over-emphasis on control (controlled experiments as nearly the only acceptable form of empirical evidence) that existed in the 1990s has disappeared; that's good." (A26)

This may be due to software engineering's roots in computer science and engineering, where traditional, controlled experiments are dominant. This bias on control may be lessening as we continue to introduce more research methods from social science, where control is not always the highest priority.

The solution-focused nature of software engineering means a bias towards realism.

Now, our community priorities appear to have shifted away from control. With the widespread adoption of empirical research techniques and increasing collaborations with industry partners, highly realistic and generalizable research is coming to the forefront. Some authors saw a bias towards realism, and this could be due to the solution-focused nature of software engineering, the need to have a practical impact, and close ties with industry members. Authors pointed out some reasons why this may be problematic, stating:

> "SE researchers are not interested in "boring" research; they are very solution-focused. As a result, fundamental research does not get done in my opinion, because it requires giving up realism for control." (A3)

> "I think in the last decade there is a growing and strong bias towards realism. It is a strength but also a weakness: bias towards short term impact, bias towards solutions that work although nobody cares why, results overfitting available data." (A42)

I elaborate on these comments further in Chapter 7.

Reviewers demanding generalizability means authors have to shift their focus.

The most pervasive opinion we found through this analysis was that there was a bias towards generalizability in the software engineering community, particularly with regards to the reviewing process. Authors perceive that generalizability is too heavily emphasized by reviewers because it is easy to criticize in papers, even if work is highly realistic or controlled (A19, 28). Authors thought this was leading reviewers to have "totally unrealistic expectations regarding generalizability" (A26) and sample size, and one author said:

> "Generalizability is highly demanded by reviewers. This is why there is an increasing number of subjects (software systems and developers) in studies over the last decade." (A34)

Thus, in order to publish, authors think that they need to include larger and larger sample sizes in their work, beyond what is really necessary, making research more costly and time-consuming to conduct.

6.8 Summary and Important Takeaways

I summarize the findings by restating the answers to the research questions, as well as identifying some key takeaways from our analysis.

6.8.1 RQ1: What research strategies and data sources are described in research published at ICSE?

We found that ICSE papers contained far more Computational Studies (76.7%) than any other research strategy. We did not find a single instance of an Experimental

Simulation within the sample, and we also found that Field Studies were very uncommon. We also found that there was a very high use of Trace Measure data (82.2%) compared to other data sources, though there were many papers that reported using Self Reports and Visible Observer data to inform their research.

We investigated the possibility that researchers were triangulating Computational Studies with Trace Measures with the use of other research strategies and data sources, and we found that only 18.9% of the papers in our sample reported more than one research strategy and 20.9% reported more than one data source. While some authors indicated that they used different forms of triangulation in their work and that it was a priority for them, 28.3% of our authors said that they did not understand the concept of triangulation in the context of research.

6.8.2 RQ2: Why do authors choose the research strategies and data sources they describe in their papers?

Mostly, authors chose the methods that were the best fit for their research question. This helps us to understand why we saw so many Computational Studies and so much Trace Measure data; most of our authors said that they studied a topic area that was technical in nature, and Computational Studies using Trace Measure data are often the most appropriate choice to answer technical research questions. Likewise, the authors who were interested in research questions and topic areas around human behavior used research strategies and data sources that included more active human involvement.

However, we also found a number of reasons for choosing research strategies and data sources that were problematic. Access to data and ease of use were some of the factors that played into authors' decisions. Authors also indicated that a lack of access to developers made it harder for them to triangulate their research through studies that involved active human participation.

6.8.3 RQ3: What is the "balance" of quality aimed for across generalizability, control, and realism in research published at ICSE?

Both the systematic mapping study and the survey indicated that ICSE community may be biased towards prioritizing realism and generalizability over control. Mostly,

authors recognized the connection between research strategy choice and generalizability, realism, and control, and prioritized the criteria that are most appropriate for their research strategy. However, as individual authors, many often prioritize realism out of a desire to have practical impart or to satisfy the demands of industry stakeholders. After prioritizing realism, authors indicated that they also needed to prioritizing having high generalizability to satisfy unreasonable reviewer demands to be accepted for publication. After prioritizing high generalizability and realism, it is unreasonable to expect an author to also produce highly controlled research.

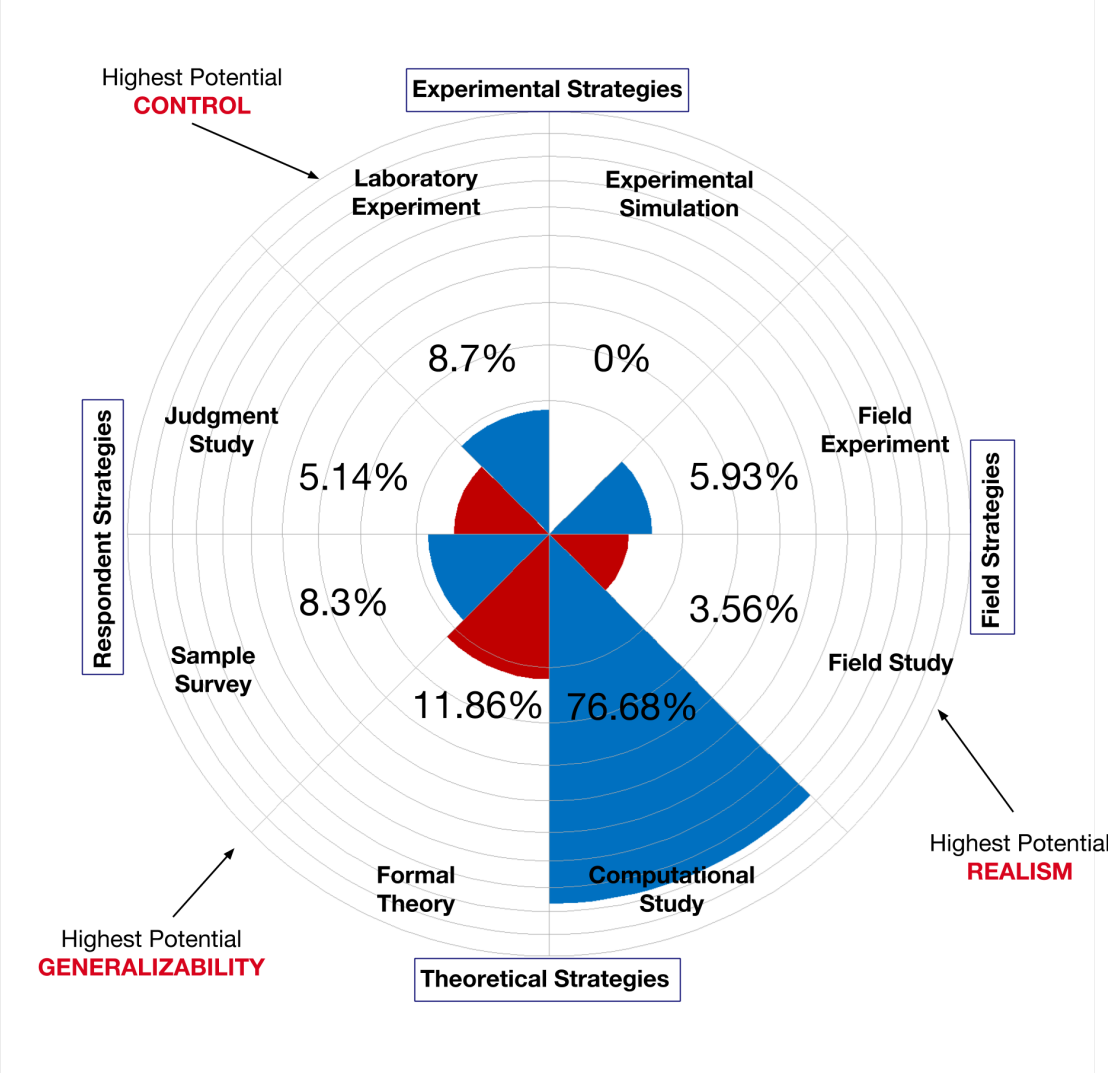


Figure 6.6: This figure shows that there appears to be a shift towards realism and generalizability, and away from control.

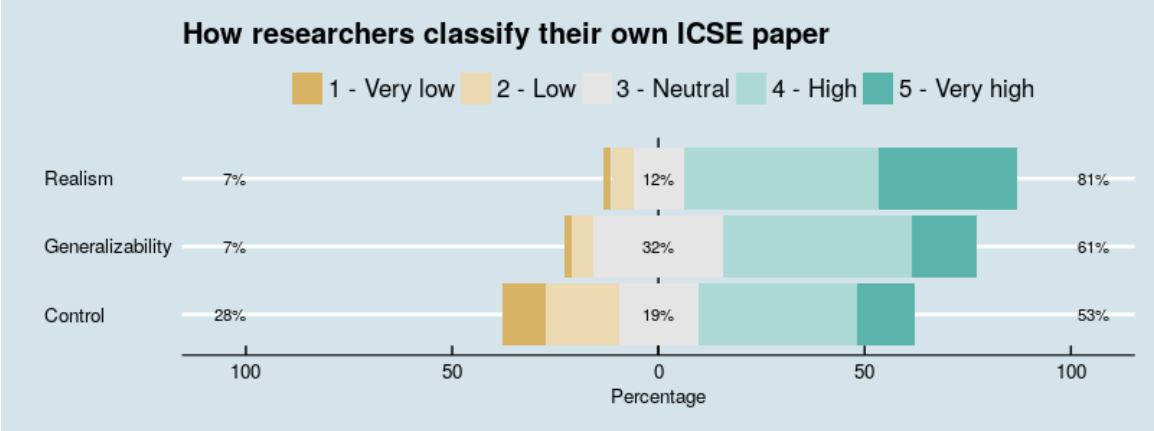


Figure 6.7: Authors indicated that their papers are higher in realism, closely followed by generalizability, and lower in control.

Chapter 7

Discussion

In this chapter I discuss the implications of our findings, identifying issues for community reflection, discussion, and action.

7.1 RQ1: What research strategies and data sources are described in research published at ICSE?

As mentioned in Chapter 6, I found a high use of Computational Studies and Trace Measures in software engineering. Using the Computational Study approach has some real benefits; it may be more efficient than other approaches, allow for the analysis of large amounts of data, and allow us to test our tools and approaches "objectively" against benchmarks that demonstrate that they improve on state of the art approaches. However, they also have their drawbacks; when using a computational approach, it is difficult to truly understand the social factors that cause the behaviours we measure. As a sociotechnical research domain, humans play a major role in many of the phenomena we study, and taking social factors into account is key to developing an understanding of these phenomena.

Like Computational Studies, Trace Measures have certain benefits and drawbacks. They show traces of human behavior that are not influenced in any way by the knowledge that the traces would be used for research purposes. Particularly in software engineering, it is often easy to access very large amounts of this type of data, making it easier for researchers to study a wide variety of issues without having to spend a significant amount of time collecting new empirical data. Similarly to Computational Studies, the major drawback to the use of this type of data is that it does not eas-

ily allow for the understanding of the social and contextual factors that led to the data's creation [2], and researchers cannot exert control on the creation of new data to observe potential changes in behavior.

While it is possible to triangulate with additional research strategies and data sources to make up for these limitations, our results indicated that this is not as common of a practice as one might think; the vast majority of the papers in our sample reported only one research strategy and source of data. Additionally, 17 out of our 60 participants indicated that they did not understand the concept of triangulation in the context of research. Triangulation is an important research concept and incredibly powerful tool for increasing aspects of research quality, and I identify this lack of triangulation knowledge within the research community as potentially problematic. While the drawbacks associated with Computational Studies and Trace Measures may be acceptable at the individual study level, I suggest that these limitations may become problematic when they are associated with a large portion of our collective research output. This issue could be mitigated by triangulating these studies with complementary research strategies and data sources, but without a strong community understanding of triangulation, this is not possible.

What should we do about it?

To address these issues, I call for action from the community. First, I suggest that we need to further investigate the role of triangulation in the software engineering research community. A replication of the methodology used in Siegmund's investigation into internal and external validity in the software engineering research community [42] to understand triangulation would be an excellent avenue for future work. Once we have an understanding of the community's understanding and perceptions of triangulation in software engineering, we will be able to identify the scope of the lack of triangulation knowledge identified in our study, and hopefully, identify the root causes and suggest possible solutions. If we find that the issue is widespread, I suggest that we change the way we educate new software engineering researchers. Incorporating lessons about triangulation into our graduate school curricula would help to increase overall awareness of the benefits of triangulation in research, and hopefully lead to new researchers using more diverse research techniques in their work.

Second, I suggest that community members should diversify the research strategies and data sources they include in their work. I do not wish to suggest that there is anything wrong with any of the ICSE papers included in our sample; however, the inclusion of more active human participation in our research, both by triangu-

lating computational work and as standalone studies, would help us to address and understand more of the social factors that affect software development. This diversification will help to increase the levels of generalizability, realism, and control in our research output as individuals, as well as the community. Additionally, it would help to strengthen our collective understanding of the social factors that influence software development.

7.2 RQ2: Why do authors choose the research strategies and data sources they describe in their papers?

I found that participants primarily chose the research strategies and data sources they used in their papers based on what they thought was the best fit for the research question; However, they were also influenced by some factors that are potentially problematic. They were tempted to make use of data sources that were publicly available and to choose methods that were efficient or easy to use. While it makes pragmatic sense to work with the data that is available to you and easy to use, I suggest that this should not compromise the approach of choosing the best methods to answer your research question. Just because data is available, or an approach is efficient, does not mean that it is the best fit to answer the research question.

Participants also faced barriers accessing developers to participate in their studies, and this prevented some of them from including active human participation in their work. This is problematic, as access to naturalistic settings and human participants is key to diversifying the research strategies and data sources we use in our work and generating knowledge about the social factors of software development.

What should we do about it?

To address this issue, I call for more members of the research community to engage in a collaborative discussion with industry to find ways to enable more collaborations to our mutual benefit. Removing barriers to participant access could empower researchers to diversify the research strategies and data sources that they use in their work, helping to alleviate the issues created by a heavy emphasis on the use of Computational Studies and Trace Measures. Additionally, I speculate that further cooperation with industry would help us to better align our research projects to solve real development problems, which was a high priority for our participants.

I also suggest that as individuals we must think more critically about the factors that influence the design of our studies. We must strive to not simply answer our research questions, but to design our studies using the techniques and data sources that will provide us with the *best* possible answers. This may not always be the easy choice, and it may require more work and resources to conduct our research this way, but the insights we produce *will* be more valuable in the end. Reviewers cannot determine whether we took the path of least resistance or chose the best possible option available from our papers alone, so the onus is on each of us to ensure that we strive to make the most appropriate choices in our studies. As one of our participants said, "just because data is convenient or available does not mean it reveals what we are looking for." (A4)

7.3 RQ3: What is the "balance" of quality aimed for across generalizability, control, and realism in research published at ICSE?

This research illuminated issues with bias in software engineering, where we are prioritizing realism and generalizability in our research over control. While we recognize that it is best to prioritize the desirable research criteria that have the best potential for maximization according to the research strategies that we are using, we prioritize realism to provide solutions to industry problems, and we prioritize generalizability because we are beholden to reviewer demands for increasingly large sample sizes.

This prioritization may have an effect on our collective research output. Our participants speculated that our current prioritization "is a strength but also a weakness: bias towards short-term impact, bias towards solutions that work although nobody cares why, results overfitting available data." (P60) By neglecting to address control through the use of a diverse set of research strategies and active human participation, we may not be conducting enough fundamental research (A3) or research in the field that would ground our solution-focused research studies in theory. I suggest that this has the potential to slow our ability to progress as a research community because understanding *why* our tools and approaches work would help us to create *better* solutions to problems much more quickly.

What should we do about it?

First, perhaps we can investigate the dissonance that exists between our priorities as paper authors and our perceptions of reviewers' priorities. Paper reviewers are authors themselves, so reviewers probably have the same preferences as authors, and some other phenomena is causing this problem. Participants already suggested that the possible cause could be that reviewers are overworked, and that generalizability really is easier to criticize in a research paper than realism or control. I believe that studying this issue could help us to identify the causes of this problem so we may address it.

I speculate that this problem is caused by a lack of time available to review papers, as well as an inability to know how to properly review every paper; there is a myriad of research methods available to a software engineering researcher, and so it is unreasonable for a reviewer to be well versed in all of them. I suggest that if these are the factors that are causing researchers to perceive that reviewers are focused on generalizability, that we must change our reviewing process to ensure that reviewers have enough time to review papers and that reviewers are only assigned papers that follow techniques that the reviewer is comfortable assessing due to their own previous experience with that type of work.

Solving this reviewing issue may further encourage researchers to include more active human participation and industry involvement in their work and diversify the research strategies and data sources reported at ICSE. This is because researchers would not feel as pressured to include incredibly large sample sizes in their studies to achieve generalizability, therefore enabling for the use of more field and experimental, and respondent strategies.

Finally, I feel we need to engage in a community discussion about our goals for the future of software engineering. I do not suggest that our bias towards realism and generalizability is a problem by itself, but I do argue that our prioritization of desirable research criteria should reflect our shared values and goals for the future of software engineering and be based around careful consideration of the benefits and drawbacks that result from these biases. If the research community believes that this prioritization reflects what we want to see for the future of software engineering, then there may be no need for drastic change. However, if we see a bias towards realism and generalizability as being in conflict with our vision for the future of our research field, then we must then also discuss the changes we need to make moving forward to shift the balance of the three desirable research criteria that aligns with our values. I suggest that perhaps we could create publication venues and conference tracks that

provide a better avenue for under-represented research that addresses social factors in software engineering, enabling researchers to conduct and publish more diverse research that will move us forward in our understanding of the sociotechnical aspects of software development.

7.4 Understanding Research from a Social Perspective

One limitation of the research lens in software engineering is its theoretical underpinning. The adapted models we used in this research may not work for all researchers in software engineering, and their applicability may depend on the worldview of each individual researcher. They should apply well to work conducted by empirical researchers in software engineering that include considerations for human factors in their work. A researcher who does not view software development as a sociotechnical process may struggle with the use of this lens. If you do not see the social aspect of software engineering as a crucial factor in understanding software development, you are unlikely to agree with the premise of discussing research *as it relates to social processes*.

For example, there are two ways of understanding realism. From the social perspective, this refers to the realism of the natural, *social context* under which data is gathered, and the applicability of the findings from this data to real-world *social processes*. From a more general or technical perspective, this means the realism of the natural context under which data is gathered and the applicability of the findings from this data to some aspect of the real world. Take, for example, a software development tool that is designed and evaluated through computational experiments using a large set of software repositories as a data source. This work would likely be very realistic in the sense that they used *real code* to design and evaluate their tool, and the tool is likely to be very applicable to other repositories of a similar type. However, this study would likely be lower in realism *from a social perspective*, since they did not account for the humans that created the code they used to design the tool, and they have not evaluated the tool in such a way that would allow them to make any claims about how applicable the tool is to *the social process of developing software*.

In this way, I suggest the majority of the Computational Studies that we observed do not actually maximize realism or generalizability with regards to the social aspects of software engineering. They may be very realistic and generalizable with respect to the behavior and nature of software, but understanding software is entirely different from understanding developers. To truly maximize *realism of social contexts*, researchers would need to conduct naturalistic Field Studies that address contextual social factors. Truly maximizing *generalizability to a population of human actors* requires conducting Sample Survey research on representative populations that lead to generalizable findings or developing theories of human behavior based on previous empirical work. While the findings of this research may suggest that we prioritize realism and generalizability over control, we may not actually be producing much work that is highly realistic or generalizable with respect to the populations, human behaviors, and social contexts associated with software development.

I suggest that as a community we should engage in a discussion about generalizability, realism, and control with regard to both technical and social factors of software development. It is important that we recognize the difference between the levels of these criteria respecting both software and developers; both are valuable for software engineering research, and both deserve equal attention in the community.

Chapter 8

Limitations

As I discussed previously throughout the thesis, no study is perfect and every methodological choice has both benefits and drawbacks for overall research quality. In this chapter, I identify limitations and threats to validity associated with this research, why the benefits of these choices justify the risks and the measures we took to mitigate potential issues and biases through the research design.

8.1 Threats to Construct Validity

As this work adapted a research lens from another domain, focused on behavioral research, and from a different time period, there are limitations to how it can be applied and understood in the software engineering research domain.

8.1.1 Complexity

In addition to the limitation of relying on a specific worldview, this research lens is very complex and may be difficult to grasp in its entirety. While it is easy to learn the research strategies and data sources at the surface level, it is difficult to understand the research lens well enough to apply it to your own research design process, and even more difficult to apply it to the work of others.

One reason it is so complex is that there are a number of details and characteristics associated with each strategy, and you must consider all of them when categorizing research. Some study designs may have characteristics that suggest it could be categorized in more than one way. For example, a Laboratory Experiment used to evaluate a tool could also include asking a number of questions to authors about their expe-

riences with the tool, much like what is done in a Judgment Study. However, the specific and controlled context of the laboratory environment and the setup of the tasks indicates that this is a Laboratory Experiment, not a Judgment Study. Rather than forming their own strategy, these questions would instead be a form of data source triangulation and a way of explaining experimental results within the Laboratory Experiment. Distinctions like this are difficult to make without first spending a lot of time learning how to properly apply the models to research papers. I believe that this contributed to the discrepancies I saw between my categorizations and the categorizations of authors discussed in Chapter 6.

There is also no common taxonomy for describing research, which further complicates the task of analyzing a research paper based on this lens. These models are not commonly used to describe a research approach, and most of the papers we analyzed did not use these terms at all or used some of the terms in a different way. For example, researchers used the term "field study" to refer to a number of different techniques at different levels of abstraction, which means that you cannot rely on author self-categorization to determine the research strategies or data sources used in a paper. To apply this research lens to software engineering it requires significant effort to properly understand the lens, which limits its ability to be widely used by the research community.

8.1.2 Purely Technical Research

As this model was created for human and behavioural research, it does not easily capture the intricacies of more technical and mathematical research. As a sociotechnical domain, technical, algorithmic, logical, and mathematical research forms a large portion of our collective research output. Our research shows that, overwhelmingly, this research is conducted using Computational Studies and Formal Theory strategies because of the technical nature of the research itself. Our research lens does not capture the intricacies and diversity of these types of studies, as it is concerned primarily with showing the differences between different strategies used to understand human behaviour in software engineering rather than technical aspects. One major limitation of this research lens is that it may not be the best choice for discussing highly technical sub-communities in software engineering. In these cases, I would recommend using models designed for technical research, either on their own or in addition to our research lens. For example, Kagdi, Collard, and Maletic [16] present

a taxonomy for describing software repository mining studies. Ambreen *et. al.* [1] conducted a systematic mapping study of primary studies in the requirements engineering community, classifying papers based on the types of methods they reported.

Many software engineering papers are solution-focused, so Wieringa's Design Science Methodology [58] could be a good choice for understanding technical research that centers around the design of a new tool or technique. For research on cloud computing, Souri, Navimipour, and Rahmani [49] present a taxonomy for verification approaches in cloud computing. For a more general taxonomy, The Guide to Advanced Empirical Software Engineering [41] provides a number of guidelines for empirical research in general that may be helpful for understanding and describing technical studies in software engineering, particularly when designing a study.

8.1.3 The Impact of Technological Advancements

The original models of research strategies and data sources were published in 1972, and reflect the reality of behavioural research at that time. Since then, the development of new technologies has changed not only the way we conduct research, but also the social contexts that exist in society. This means that there are some fringe cases where new technologies create situations that may challenge the use of the models when understanding and classifying research.

One example is the use of new technologies by researchers to gather participant data. When the original models were created, it was much more difficult to gather Self Reports or observational data from human participants for research purposes than it is now. Researchers had to recruit participants, distribute surveys, and conduct interviews, experiments, and observations in person, through the mail, or over the phone. Conducting research this way is time-consuming and expensive, and this can limit a research project's scope due to resource constraints. Now, researchers can send online questionnaires to participants via email and conduct face-to-face interviews through video calling services such as Skype and Google Hangouts. This is the same for many forms of participant observation as well, where software engineering researchers can use tools for eye tracking, mouse tracking, and video streaming to observe participants and their development behaviour remotely. All these technologies mean that researchers can now conduct research "in the field" without ever entering the building, but make it more complicated to distinguish between Field Strategies

and Experimental or Respondent Strategies because the researcher no longer has to physically enter a natural environment to follow a Field Strategy.

The adoption of distributed software development is another technological advancement that complicates the classification and understanding of social research in software engineering. In these contexts, developers are distributed across various locations, communicate through a series of technologies such as email and online team communication channels like Slack, and conduct their work remotely. This means there are two "natural settings" for a developer in a distributed team; their physical environment and the team's virtual workspace. A researcher studying this population could study a distributed team's physical setting, observing how individual developers conduct their work in their homes or offices. Alternatively, a researcher could study their virtual environment by observing the team's communication and development tools. Virtual communities did not exist when the original models were created, so the definition of "natural setting" is complicated when applying them to modern research.

We addressed these issues in our adaptation of the models. Initially, we intended to apply the original circumplex model as literally as possible, considering a natural setting to be a physical place, as it was not possible to have a virtual setting or conduct field research remotely when the models were designed. As discussed above, this view was limiting and did not accurately capture the realities of research that is heavily influenced by these technological advancements. We then changed our adaptation to reflect the spirit of the models rather than the literal definitions and broadened our description of "natural setting" to include virtual environments, and allow for researchers to gather "field" data remotely.

This is a limitation of our adaptation of the circumplex model, as it is one interpretation of the model and has its drawbacks. There is a strong case for why collecting participant data remotely should not be considered the same as in-person observation and self-reporting. For example, when interviewing participants remotely, a researcher might not be able to observe and note contextual factors about the participant's setting that might be influencing their responses, intricacies in tone and body language could be lost through poor connectivity, or the researcher may not be able to establish a relationship with the participant as easily as they could in person. This would hurt the researcher's ability to understand important contextual details that are necessary to maximize realism in a Field Strategy. However, there are upsides to the use of this technology that can improve the quality of fieldwork conducted

remotely. For example, a researcher conducting interviews remotely may find that they are able to interview a more diverse and larger group of participants by removing the resource constraints associated with physically meeting with a participant.

Another new technological advancement that complicates the use of this lens is the proliferation of "big data" resources. When the models were originally created, Trace Measures and Archival Records typically weren't digitized, nor were very large amounts of data available that could be easily analyzed as part of a research project. Now, aspects of human behaviour are constantly recorded by various organizations, leading to the creation of massive data resources that can be analyzed for the purposes of research. We also have far more powerful computing resources than in previous decades, and new analysis techniques and algorithms that allow researchers to conduct complex analyses of big and thick data relatively cheaply compared to traditional manual analysis. These data sources often contain large amounts of contextual data, allowing researchers to harness big data to achieve significantly higher realism in Computational Studies than ever before. This means that when applying the research lens, Computational Studies that analyze big data about human behaviour may have a higher potential for realism than other Computational Studies, and they can achieve that realism by harnessing the power of thick data to include rich contextual information as part of their findings.

In applying these models to software engineering, we must be careful when we discuss research that studies social contexts that are heavily influenced by technological advancements and consider the benefits and drawbacks to using new technologies to conduct our research. Using technologies can help us to reduce resource constraints that could keep us from studying important phenomena in software engineering, but they have limitations compared to in-person research methods that should be carefully considered.

8.1.4 Miscommunications

During the analysis of the survey, particularly the analysis of author classification of research papers, it became clear that the research lens was not adequately communicated to all the authors. This was not unexpected, as I understood that the research lens is very complex and it would be difficult to convey to authors in a succinct manner. It was unreasonable to expect that authors would read the entire description of the research lens to participate in the survey, so I had to abbreviate the descrip-

tion of each research strategy and data source down to its most basic and defining characteristics, as well as create a simple description of each of the three desirable research criteria. This introduced the possibility of confusion because there may not have been enough details for some authors to properly understand the research lens before responding to the questions. This was particularly apparent with Experimental Simulations getting confused with Computational Studies and Formal/Theoretical as a data source being mistaken for a term to describe work that had an algorithmic component.

The terms used in the research lens are not entirely new, and the terms may already have had meaning for authors and introduced further confusion. For example, authors were likely to go into the survey with their own understanding of "control" in the context of research. These preconceived notions may have affected the authors' perception of the terms, even after having been provided with definitions. This seemed apparent with regards to control; we described it explicitly as having to do with control over *measurement of human behaviour*, but some authors appeared to understand it as control over measurement of any behaviour; human or software. These misconceptions are a limitation of the survey because it introduces doubt as to whether we correctly understood author responses in our analysis.

To mitigate this issue we conducted member checking, which proved to be very helpful and allowed us to confirm our findings without reliance on the terms in the research lens. However, it should be noted as a limitation that some of the original quotations from the survey may reflect a different understanding of the research lens.

8.2 Threats to Internal Validity

As many of the tasks associated with this research were conducted alone by a single researcher, there are few threats to internal validity associated with different forms of bias.

8.2.1 Potential for Researcher Bias

One limitation of this study is the potential for researcher bias during the paper classification process, as I classified research papers alone. In anticipation of this issue, we took some measures to mitigate this possibility. First, developing the research lens was a team effort, where we explored and discussed the papers, often reading the same

paper together to understand how to best apply the models to software engineering. Through this process, we attempted to ensure that my understanding of the research lens reflected our shared understanding and that I would classify the papers in the same way as other team members. I completed the paper classification task without any reference to previous classifications to ensure that I was not biased by my initial classification and considered only the paper itself and a shared understanding of the research lens.

We further mitigated potential research bias by including author classifications as a means of calculating a form of inter-rater reliability. This was in order to determine if there was a shared understanding between myself and the authors of the papers, who were likely to know and understand the work best. However, through the survey, it became clear that issues of miscommunication and other factors made it difficult to determine how well my classifications actually aligned with author perception of their own research. I conducted member checking wherever possible to determine if there was a common understanding of the work (without reliance on the research lens) presented in the paper that was shared by myself and the author, in which case my classification would be considered valid. This practice would have been more valuable if it were possible to create a shared understanding of the research lens between myself and the authors, and have the authors re-classify their work, as that would be a more objective way to determine agreement.

This classification was not free from the potential for researcher bias, but we took a series of steps to reduce this possibility. It may be a useful exercise in the future for another researcher to conduct a replication of the systematic mapping study to see if our disagreements suggest an inherent bias in my work.

8.2.2 Author Perception

As with any survey, this survey has certain limitations associated with author perception. First, the survey's sample may be biased by an over-sampling of authors with an "axe to grind" with the research community. Most authors seemed to indicate that they perceived some sort of issue with the research community in the latter part of the survey, so this may have been the case. Oversampling of extremes (highly positive opinions, highly negative opinions) is a common limitation to surveys. In this case, an oversampling of researchers with "issues" to discuss may have helped

to identify problems within the community that may not have been clear had the sample included more neutral authors.

Additionally, the survey's findings may be biased because authors are unlikely to respond in a way that reflects poorly upon them. This is another common limitation of surveys and may have affected the responses to a few questions in particular. For example, when asked about the levels of each of the three criteria in their research papers, authors responded positively on average, with few authors saying that their paper was "low" or "very low" in any of the three criteria. We speculate that authors may have also overestimated their use of certain research strategies over the previous five years of their research career in an effort to appear experienced in a more diverse set of research strategies. The findings should be considered limited to the responses of authors *relative* to their responses for other categories in the same question, rather than the levels of the individual responses themselves.

8.3 Threats to External Validity

As this research focused on a particular sub-community of the software engineering research domain, there are a number of potential threats to external validity associated with this work.

8.3.1 Sample Selection

One limitation to the systematic mapping study is the selection of just three years of ICSE technical track papers as the sample for classification. Choosing a different sample likely would have produced different results, and below I discuss how a different research sample would have affected the findings.

First, by selecting only three years of proceedings, it is not possible to show trends over time in the research community. The sample contained 253 papers, which was a large sample of research papers for analysis. Had I analyzed more years of proceedings, the findings may have shown that the research community is diversifying its use of research strategies and data sources over time to become more balanced, rather than just suggesting that there is currently an imbalance. There were a number of reasons for doing this:

- It was not our intent to suggest anything about how we may be changing as a community over time, but rather to show our current reality.

- Classifying 253 papers was already a very time intensive activity that could not be rushed for fear of missing crucial details.

- Integrity of author memory and author access; the further we went back into historical proceedings, the higher likelihood that authors would no longer be available, such as a student author getting a job as a developer in industry and no longer providing a current public email address. Additionally, earlier years of ICSE proceedings reflect work that happened further in the past, and author could forget the details of their work when classifying their papers. This would have caused issues with our validation phase for the systematic mapping study, as it would have introduced doubt as to the integrity of author classifications.

Second, there is a limitation associated with the inclusion of only technical track research papers. ICSE has a number of tracks, such as New Ideas and Emerging Results (NIER), Software Engineering in Practice (SEIP), and Software Engineering in Society (SEIS). These tracks may have produced a different and more varied distribution of research strategies and data sources. For example, NIER calls for new ideas and welcomes research that employs strategies and methods that are less common. SEIP and SEIS also focus more on the human aspect of software engineering, and how tools help practitioners and affect broader society, and may have included more human-involved research strategies and data sources. This means that the inclusion of additional tracks of ICSE may show a difference in the findings towards the inclusion of more active human involvement in research.

Third, there is a limitation associated with the choice of ICSE as our target venue. Similarly to choosing other research tracks for our sample, our findings likely would have changed if we had selected a different publication venue. Anecdotally, it has been mentioned to the research team by other members of the community that Foundations of Software Engineering (FSE) may have proved to be more imbalanced towards Computational Studies and Trace Measures, where venues such as Empirical Software Engineering and Measurement (ESEM) may have been more distributed and contain more examples of work that involved active human participation. This is a limitation to our work, and it is important to note that the findings from the systematic mapping study are limited to the ICSE community.

Finally, there is a limitation associated with selecting only papers that had been accepted for publication. It would have been interesting to see if there is a correlation between choice of research strategy or data source and rate of publication, but this

data was not available to us. This work reflects only research that was accepted for publication, not the broader context of research that was conducted and submitted regardless of acceptance or rejection.

8.3.2 Participant Selection

We focused on contacting first authors and only invited subsequent authors to participate if the authors before them in the author list could not be contacted, which may have introduced a bias into our sample. First authors are typically the researcher who conducted the majority of the work, which is often a graduate student. This is reflected in our participant population, where around half the authors indicated that they were university students at the time they conducted their research. This may have introduced bias, as students may have been less exposed to the software engineering research community, and thus have less experience on which to base their responses to the survey. Students may also be less likely than more experienced researchers to understand triangulation, which could have biased the responses towards misunderstandings of triangulation.

While this may be a limitation of the survey, this participant selection technique was the best choice given the alternatives. It would have been inappropriate to invite every author of each paper to participate in the survey, as it would have introduced multiple (possibly contradictory) perceptions of the same paper, which would have complicated our ability to compare my classification of the research papers with that of the authors. Additionally, many authors published multiple times at ICSE, with more prolific researchers appearing more than five times in three years of proceedings. It would be unrealistic to ask an author to respond to the survey for five separate papers, and it would introduce ethical concerns by bothering authors with repetitive email invitations. It was also not possible to assume, other than the first author, who would be best qualified to answer questions about the research, as there are different conventions and preferences associated with author ordering. While there is a limitation associated with primarily inviting first authors to participate in the survey, it should be noted that alternative choices likely would have been more problematic.

I also recognize that my status as a student with a supervisor that is highly regarded within the research community may have influenced some authors to participate in the study as a favor to her, or that their responses may have been influenced to impress her; likewise, it is also possible that some authors declined to partici-

pate based on their opinion of my supervisor, or were negatively influenced in their responses based on their relationship.

8.3.3 Excluded Viewpoints

As discussed in Section 8.3.1, we did not look at rejected papers in our systematic mapping study, and thus we did not include authors whose papers were rejected from ICSE during the same time period. This may have impacted the findings from the survey because these authors may have very different opinions about research community bias, especially because their work was rejected. These authors also may have been able to bring insights to the findings about the types of reviews they received on their work, and why they perceive that their work was rejected. This would have been a very valuable piece of analysis to include in our work, however, it was not possible to contact these authors to participate, as information about rejections and rejected papers is not available for ICSE.

Another viewpoint that we did not include in the survey is that of the paper reviewers. The survey asks authors about their priorities when conducting research and their perception of the community. We also show work that was published, but not work that was rejected. This places limitations on this research in terms of understanding what can, and what cannot be inferred about reviewer priorities from the findings. Authors *perceive* that reviewers are focused on generalizability, however, reviewers of papers could be much more concerned with other issues when they review papers. The distribution of research strategies and data sources used in the papers that were accepted may or may not reflect reviewer priorities; without asking the program committees from the years that we sampled, it is not possible for us to make inferences about what reviewers actually value when it comes to accepting and rejecting research papers. This suggests that sampling reviewers and rejected authors and investigating these issues would be a good avenue for future work.

8.4 Addressing Limitations in Existing Work

As discussed in Chapter 3, Stol and Fitzgerald [50] also applied Runkel and Mcgrath's circumplex model to software engineering, but there were some limitations to their work. Our research addresses some of the issues and limitations of their paper by

addressing the topic from a sociotechnical perspective and extending the work through the use of a systematic mapping study and a survey.

Stol and Fitzgerald's interpretation of the circumplex model, in its application to software engineering, considers that an "actor" could be more than just a person or social entity; Stol and Fitzgerald detail how the actor under study could be either a human/social entity or a software system. They describe how one could conduct Sample Surveys or a Field Study on a software system, saying that researchers could conduct a data mining study of repositories and that this would fall under Sample Survey. This is an important distinction, as the original circumplex model was designed for human and behavioral research, in which case the actors would *have* to be people or other social entities.

This has implications for the rest of the work; the three desirable research criteria in this work refer to both social and technical contexts simultaneously, rather than from a purely social perspective like the original model. In their work, a Sample Survey of GitHub repository mining would be considered highly generalizable as it would likely generalize well to other repositories, but this says nothing about the social development contexts that resulted in the creation of the repositories themselves. From a sociotechnical perspective, this could be problematic, as it allows for researchers to abstract the developer out of a software development context and make the same claims about aspects of research quality as a study that considers developer behavior and perception.

To address this limitation, we established that actors could only be human or social entities, meaning that our lens shows a clear difference between strategies where humans are included and instances where human participation is not part of the research design. Stol and Fitzgerald also do not include any other complementary models from Runkel and McGrath's work, like their description of different data sources and their benefits and drawbacks when included as part of a research design. We included their model for understanding sources of data, as we found it was another complementary measure to show what types of human participation were included as part of the research strategies, as each of these also has its own benefits and limitations.

Additionally, there are some limitations to their paper. First, the authors only show examples of papers that fit into each research strategy. They do not attempt to categorize a large body of work, such as the proceedings of a conference, to show how well the model applies in practice and how to handle edge cases that may challenge

the model. They also do not consider secondary studies such as systematic literature reviews in their model. This means that another researcher attempting to apply the lens to a large body of work, or their own research, may encounter difficulties with edge cases that don't fit into the circumplex without additional guidance from the research team. We addressed this limitation by including secondary studies as part of the model, as well as applying the model to ICSE technical track papers without exclusion of any work and discussing edge cases and situations which challenge the model so that it may be applied more easily to software engineering research by others in the future.

Another limitation to their work is that they do not consider that the research strategies only *help* researchers to maximize certain criteria; For example, the paper states that "field studies achieve a 'maximum' in realism of context" when in reality they simply provide a research team with the *opportunity* to maximize realism. The research design that follows that strategy must allow for realism to truly be prioritized. To address this limitation, we discussed our interpretation of the models in a way that made clear that the strategies only provided researchers with an opportunity to maximize a criterion rather than with an automatic maximization. Our survey also further investigates this issue by exploring the levels of each criterion that authors *perceive* in their papers, as well as how they prioritize these criteria and their perception of the research community's prioritization of the criteria. This showed us that there was a relationship between choice of research strategy and criteria prioritization, and allowed us to highlight issues with prioritization that helped to suggest some reasons why we may choose certain research strategies over others.

Chapter 9

Conclusion

In this chapter, I suggest possible avenues for future work, as well as conclude the thesis by reiterating the important takeaways from this research.

9.1 Future Work

As this research investigated a number of issues in the software engineering research community and had some limitations, there are a number of possible areas for future work and follow-up studies.

As I discussed in Chapter 8, one of the major limitations to this work is that it is only applicable to the ICSE community. It would be a useful exercise to conduct follow-up studies with other software engineering venues. Our interview study research suggests that replicating this study with a software engineering journal, such as Transactions on Software Engineering (TSE), or the ESEM or FSE communities would lead to different results. Not only would a replication study in software engineering give us a means for comparison of the types of work that are published in different communities, but it would identify additional potential fringe cases challenging the research lens that would help us adapt our descriptions to be more reflective of the work published in software engineering. Replication within another sociotechnical research domain, such as HCI, would be incredibly beneficial to this research as well. Though it would require developing new descriptions of the research lens to reflect the reality of that domain, it would help to show similarities and differences in the research strategies used between the two research areas and potentially indicate if this lens could be used as a common taxonomy for describing sociotechnical research,

which would be helpful for researchers whose work spans more than one sociotechnical research community.

The survey indicated that there was a broad spectrum of understanding of triangulation in software engineering, with many authors indicating that they had never heard of the term before. This was just one of the questions in the survey, and so we did not gather the data necessary to fully understand this issue. Authors who gave simplistic definitions of triangulation may actually understand the complexities surrounding the different types of triangulation that can be used in research. Authors who indicated they had never heard of the term may actually have a depth of knowledge of the practices of triangulation and use them frequently in their research without knowing the term for these practices. We also did not investigate this in the systematic mapping study beyond indicating the number of papers that utilized more than one research strategy or data source to accomplish their research goals. Since triangulation is an important concept in any research domain, I suggest that additional research should be conducted to understand triangulation in software engineering to determine the severity of this issue, the effects it might be having on our collective body of work, and potential solutions.

This study led to a number of interesting findings, and the authors indicated a series of issues that they perceived in the software engineering community. For example, authors suggested that there are problems accessing industry authors for research studies and that reviewers are overworked and overprioritizing generalizability. However, the identification of these issues was based on the responses from a small sample of authors; we don't know how widespread these phenomena are in software engineering as a whole. As mentioned in Chapter 7, a follow-up survey on the broader software engineering research community to determine the severity of these issues and to identify possible solutions would be a very powerful avenue for future research.

9.2 Conclusion

As a sociotechnical research domain, software engineering encompasses both social and technical factors of software development. These intricacies make it imperative for the software engineering research community to include a variety of both social and technical research methods to study the complexities of software development. Motivated by observing a lack of active human involvement in software engineering

research papers, my goal was to investigate the diversity of research methods and human involvement techniques in the software engineering research community and its impact on research quality. To accomplish this, I applied Runkel and McGrath's models to three years of ICSE proceedings to investigate the distribution of research strategies and data sources reported in these papers.

The first major contribution of this thesis is our adaptations of the circumplex model of research strategies and the data sources model to the software engineering domain. This was a team effort, where we iteratively classified papers and adapted the models to ensure that they accurately reflected the reality of modern software engineering research papers. I then applied these models to three years of ICSE proceedings in a systematic mapping study, producing classifications for each of the papers according to the models. This study showed that 76.7% of papers reported a Computational Study and 82.2% of papers reported the use of Trace Measure data, far more than any of the other research strategies or data sources. This imbalance suggested a potential prioritization of realism and generalizability over control.

To validate these startling findings, as well as investigate what factors may have caused researchers to make these choices, I surveyed the authors of these papers. I found that these decisions were influenced by a number of factors. Researchers chose the methods that were the best fit for their research questions. At the same time, they were also hindered by difficulties accessing developer populations and tempted to use easily accessed data and simple, efficient research strategies. I also found that the potential imbalance in research criteria suggested by the systematic mapping study reflected reality; authors indicated that their papers were higher overall in generalizability and realism than control and that they actively prioritized these criteria out of a desire to satisfy industry collaborators, impact real software development, and meet the demands of the reviewers who act as the gatekeepers to publication. We were also alarmed by the number of authors who indicated that they had no knowledge of triangulation in the context of research, as triangulation is a crucial concept for understanding how to balance the tradeoffs of generalizability, realism, and control in research.

These findings have a number of implications for the software engineering research community. An imbalance in prioritization of realism and generalizability over control means that our collective body of work may not adequately address control of measurement of behavior over extraneous factors. Research that maximizes control helps us to understand the causal factors of behaviors in software development, giving

us the knowledge needed to design better tools and approaches and help to create theories of software development behaviors. This means that a bias against control could perhaps slow the overall progression of software engineering research as a whole.

Therefore, I call the software engineering research community to action. This research shines a light on a number of possible issues in our community that need to be further investigated, discussed, and if deemed necessary, acted upon. Our reliance on Computational Studies and Trace Measure data leaves our collective body of work with some serious disadvantages; this type of work may not allow us to truly understand the factors that influence human behavior in software development, and I suggest that this needs to change as we move into the future. We have recognized that understanding social factors in software development is key to advancing the discipline since the first software engineering conference 50 years ago [28], and yet we still do not often actively include humans in our studies.

I hope that this thesis sparks a change within the research community and that we begin to diversify our research choices to include more active human involvement in our work. The thing that I love the most about the software engineering research community is that we constantly strive to be better. We are always looking for new techniques to bring in, different perspectives to integrate into our work, and reflecting on ourselves to find ways to improve; this call to action is no exception. Aligning our involvement of humans in our work with our shared priorities is not the first step the software engineering research community has taken in its journey as a discipline, and it will not be the last. I am confident the discussion and action that follows this work could make a lasting and positive impact on the future of software engineering.

Bibliography

[1] Talat Ambreen, Naveed Ikram, Muhammad Usman, and Mahmood Niazi. Empirical research in requirements engineering: trends and opportunities. *Requirements Engineering*, 23(1):63–95, Mar 2018.

[2] Jorge Aranda and Gina Venolia. The secret life of bugs: Going past the errors and omissions in software repositories. In *Proceedings of the 31st International Conference on Software Engineering*, ICSE '09, pages 298–308, Washington, DC, USA, 2009. IEEE Computer Society.

[3] V. R. Basili. The role of experimentation in software engineering: past, current, and future. In *Proceedings of IEEE 18th International Conference on Software Engineering*, pages 442–449, Mar 1996.

[4] Victor R. Basili. *The Role of Controlled Experiments in Software Engineering Research*, pages 33–37. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[5] Stephen M. Blackburn, Amer Diwan, Matthias Hauswirth, Peter F. Sweeney, José Nelson Amaral, Tim Brecht, Lubomír Bulej, Cliff Click, Lieven Eeckhout, Sebastian Fischmeister, Daniel Frampton, Laurie J. Hendren, Michael Hind, Antony L. Hosking, Richard E. Jones, Tomas Kalibera, Nathan Keynes, Nathaniel Nystrom, and Andreas Zeller. The truth, the whole truth, and nothing but the truth: A pragmatic guide to assessing empirical evaluations. *ACM Trans. Program. Lang. Syst.*, 38(4):15:1–15:20, October 2016.

[6] Frederick P. Brooks, Jr. *The Mythical Man-month (Anniversary Ed.)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.

[7] Kathy Charmaz. *Constructing grounded theory : a practical guide through qualitative analysis*. Sage Publications, London; Thousand Oaks, Calif., 2006.

[8] Parmit Chilana, Christina Holsberry, Flavio Oliveira, and Andrew Ko. Designing for a billion users: A case study of facebook. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '12, pages 419–432, New York, NY, USA, 2012. ACM.

[9] Reidar Conradi and Alf Inge Wang. *Empirical Methods and Studies in Software Engineering: Experiences from Esernet*. Springer-Verlag, Berlin, Heidelberg, 2003.

[10] John W. Creswell. *Research Design*. Sage Publications, second edition, 2003.

[11] John W Creswell. *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications, 2013.

[12] Steve Easterbrook, Janice Singer, Margaret-Anne Storey, and Daniela Damian. *Selecting Empirical Methods for Software Engineering Research*, pages 285–311. Springer London, London, 2008.

[13] Andreas Höfer and Walter F. Tichy. *Status of Empirical Research in Software Engineering*, pages 10–19. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[14] Ronggui Huang. Rqda: R-based qualitative data analysis. r package version. `http://rqda.r-forge.r-project.org/`.

[15] Mikkel Rønne Jakobsen. Interaction and visualization techniques for programming. In *Proceedings of the 11th IFIP TC 13 International Conference on Human-computer Interaction - Volume Part II*, INTERACT'07, pages 598–603, Berlin, Heidelberg, 2007. Springer-Verlag.

[16] Huzefa Kagdi, Michael L. Collard, and Jonathan I. Maletic. A survey and taxonomy of approaches for mining software repositories in the context of software evolution. *J. Softw. Maint. Evol.*, 19(2):77–131, March 2007.

[17] Melanie Kellar, Derek Reilly, Kirstie Hawkey, Malcolm Rodgers, Bonnie MacKay, David Dearman, Vicki Ha, W. Joseph MacInnes, Michael Nunes, Karen Parker, Tara Whalen, and Kori M. Inkpen. It's a jungle out there: Practical considerations for evaluation in the city. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, pages 1533–1536, New York, NY, USA, 2005. ACM.

[18] Barbara Kitchenham. *Empirical Paradigm – The Role of Experiments*, pages 25–32. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[19] Barbara Kitchenham, O. Pearl Brereton, David Budgen, Mark Turner, John Bailey, and Stephen Linkman. Systematic literature reviews in software engineering a systematic literature review. *Information and Software Technology*, 51(1):7 – 15, 2009. Special Section - Most Cited Articles in 2002 and Regular Research Papers.

[20] Barbara A. Kitchenham, Tore Dyba, and Magne Jorgensen. Evidence-based software engineering. In *Proceedings of the 26th International Conference on Software Engineering*, ICSE '04, pages 273–281, Washington, DC, USA, 2004. IEEE Computer Society.

[21] Barbara A. Kitchenham and Shari L. Pfleeger. *Personal Opinion Surveys*, pages 63–92. Springer London, London, 2008.

[22] Andrew Jensen Ko, Thomas D. LaToza, and Margaret M. Burnett. A practical guide to controlled experiments of software engineering tools with human participants. *Empirical Software Engineering*, 20(1):110–141, 2015.

[23] Jyrki Kontio, Johanna Bragge, and Laura Lehtola. *The Focus Group Method as an Empirical Tool in Software Engineering*, pages 93–116. Springer London, London, 2008.

[24] Anselm L. Strauss and Juliet Corbin. Basics of qualitative research: Grounded theory procedures and techniques / a. strauss, j. corbin. 01 1990.

[25] Jonathan Lung, Jorge Aranda, Steve M. Easterbrook, and Gregory V. Wilson. On the difficulty of replicating human subjects studies in software engineering. In *Proceedings of the 30th International Conference on Software Engineering*, ICSE '08, pages 191–200, New York, NY, USA, 2008. ACM.

[26] Wendy E. Mackay and Anne-Laure Fayard. Hci, natural science and design: A framework for triangulation across disciplines. In *Proceedings of the 2Nd Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques*, DIS '97, pages 223–234, New York, NY, USA, 1997. ACM.

[27] Joseph E. McGrath. Human-computer interaction. chapter Methodology Matters: Doing Research in the Behavioral and Social Sciences, pages 152–169. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995.

[28] Peter Naur and Brian Randell. Software engineering: report on a conference sponsored by the nato science committee, garmisch, germany, 7th to 11th oct. 1968". Brussels, 1968. NATO Scientific Affairs Division.

[29] Michelle OReilly and Nicola Parker. unsatisfactory saturation: a critical exploration of the notion of saturated sample sizes in qualitative research. *Qualitative Research*, 13(2):190–197, 2013.

[30] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, EASE'08, pages 68–77, Swindon, UK, 2008. BCS Learning & Development Ltd.

[31] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017.

[32] Per "Runeson and Martin" Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131, Dec 2008.

[33] Philip J. Runkel and Joseph E. McGrath. *Research on Human Behavior*. Holt, Rinehart, and Winston, Inc., 1972.

[34] I. Salman, A. T. Misirli, and N. Juristo. Are students representatives of professionals in software engineering experiments? In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 1, pages 666–676, May 2015.

[35] Carolyn B. Seaman. *Qualitative Methods*, pages 35–62. Springer London, London, 2008.

[36] Michael Sedlmair, Petra Isenberg, Dominikus Baur, and Andreas Butz. Information visualization evaluation in large companies: Challenges, experiences and recommendations. *Information Visualization*, 10(3):248–266, July 2011.

[37] H. Sharp, Y. Dittrich, and C. R. B. de Souza. The role of ethnographic studies in empirical software engineering. *IEEE Transactions on Software Engineering*, 42(8):786–804, Aug 2016.

[38] Helen Sharp. *Social and Human Aspects of Software Engineering*, pages 40–40. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[39] Mary Shaw. Writing good software engineering research papers: Minitutorial. In *Proceedings of the 25th International Conference on Software Engineering*, ICSE '03, pages 726–736, Washington, DC, USA, 2003. IEEE Computer Society.

[40] Ben Shneiderman. *Software Psychology: Human Factors in Computer and Information Systems (Winthrop Computer Systems Series)*. Winthrop Publishers, 1980.

[41] Forrest Shull, Janice Singer, and Dag I.K. Sjøberg. *Guide to Advanced Empirical Software Engineering*. Springer-Verlag, Berlin, Heidelberg, 2008.

[42] Janet Siegmund, Norbert Siegmund, and Sven Apel. Views on internal and external validity in empirical software engineering. In *Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEE International Conference on*, volume 1, pages 9–19. IEEE, 2015.

[43] Janice Singer, Susan E. Sim, and Timothy C. Lethbridge. *Software Engineering Data Collection for Field Studies*, pages 9–34. Springer London, London, 2008.

[44] Dag Sjberg, Bente Anda, Erik Arisholm, Tore Dyb, Magne Jrgensen, Amela Karahasanovic, Espen F. Koren, and Marek Vokc. Conducting realistic experiments in software engineering, 02 2002.

[45] D. I. K. Sjoberg, T. Dyba, and M. Jorgensen. The future of empirical methods in software engineering research. In *Future of Software Engineering, 2007. FOSE '07*, pages 358–378, May 2007.

[46] Dag I. K. "Sjøberg, Bente Anda, Erik Arisholm, Tore Dybå, Magne Jørgensen, Amela Karahasanović, and Marek" Vokáč. *"Challenges and Recommendations When Increasing the Realism of Controlled Software Engineering Experiments"*, pages 24–38. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[47] Dag I. K. "Sjøberg, Tore Dybå, Bente C. D. Anda, and Jo E." Hannay. *Building Theories in Software Engineering*, pages 312–336. Springer London, London, 2008.

[48] Dag I. K. Sjoberg, Jo E. Hannay, Ove Hansen, Vigdis By Kampenes, Amela Karahasanovic, Nils-Kristian Liborg, and Anette C. Rekdal. A survey of controlled experiments in software engineering. *IEEE Trans. Softw. Eng.*, 31(9):733–753, September 2005.

[49] Alireza Souri, Nima Navimipour, and Amir Rahmani. Formal verification approaches and standards in the cloud computing: A comprehensive and systematic review. 12 2017.

[50] Klaas-Jan Stol and Brian Fitzgerald. A holistic overview of software engineering research strategies. In *Proceedings of the Third International Workshop on Conducting Empirical Studies in Industry*, CESI '15, pages 47–54, Piscataway, NJ, USA, 2015. IEEE Press.

[51] Klaas-Jan Stol, Paul Ralph, and Brian Fitzgerald. Grounded theory in software engineering research: A critical review and guidelines. In *Proceedings of the 38th International Conference on Software Engineering*, ICSE '16, pages 120–131, New York, NY, USA, 2016. ACM.

[52] C. Theisen, M. Dunaiski, L. Williams, and W. Visser. Writing good software engineering research papers: Revisited. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pages 402–402, May 2017.

[53] Muhammad Usman, Ricardo Britto, Jrgen Brstler, and Emilia Mendes. Taxonomies in software engineering: A systematic mapping study and a revised taxonomy development method. *Information and Software Technology*, 85:43 – 59, 2017.

[54] Norman G. Vinson and Janice Singer. *A Practical Guide to Ethical Research Involving Humans*, pages 229–256. Springer London, London, 2008.

[55] Robert J. Walker, Elisa L. A. Baniassad, and Gail C. Murphy. An initial assessment of aspect-oriented programming. In *Proceedings of the 21st International*

Conference on Software Engineering, ICSE '99, pages 120–130, New York, NY, USA, 1999. ACM.

[56] Gerald M. Weinberg. *The Psychology of Computer Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1985.

[57] Brian Whitworth. Virtual communities: concepts, methodologies, tools and applications, 01 2011.

[58] R. Wieringa. Design science methodology: principles and practice. In *2010 ACM/IEEE 32nd International Conference on Software Engineering*, volume 2, pages 493–494, May 2010.

[59] Claes Wohlin and Aybüke Aurum. Towards a decision-making structure for selecting a research design in empirical software engineering. *Empirical Softw. Engg.*, 20(6):1427–1455, December 2015.

[60] Claes Wohlin, Martin Höst, and Kennet Henningsson. *Empirical Research Methods in Software Engineering*, pages 7–23. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[61] Claes Wohlin, Per Runeson, Martin Host, Magnus C. Ohlsson, Bjrn Regnell, and Anders Wessln. *Experimentation in Software Engineering*. Springer Publishing Company, Incorporated, 2012.

[62] Robert K. Yin. *Case Study Research: Design and Methods (Applied Social Research Methods)*. Sage Publications, fourth edition. edition, 2008.

[63] Marvin V. Zelkowitz. *Techniques for Empirical Validation*, pages 4–9. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

Appendices

Appendix A

Survey Questions

A.1 A Survey of Method Choice in Software Engineering Research

We are Courtney Bornholdt, Alexey Zagalsky, Eirini Kalliamvakou, and Margaret-Anne Storey from the University of Victoria in Canada. We'd be grateful if you could help us understand what choices you make in your research and your perceptions about our research community! Filling the survey should take 15-20 minutes.

This research is focused on analyzing modern software engineering research through a socio-technical lens. We are analyzing ICSE technical track papers to determine what research strategies and data collection methods are reported to gain a deeper understanding of human involvement in software engineering research. With the current survey, we are also interested in why researchers make these decisions, the potential impact of these decisions, and perceptions about these issues within the community.

This is a purely academic research project with no commercial interests. We will openly publish the results so everyone can benefit from them (for example, our previous study on how educators use Github - https://goo.gl/ZE326U), but will anonymize everything before doing so. We will handle your responses confidentially. If at some point during this survey you want to stop, you're free to do so without any negative consequences.

Please be advised that this research study includes data storage in the U.S. As such, there is a possibility that information about you that is gathered for this research study may be accessed without your knowledge or consent by the U.S. government in compliance with the U.S. Patriot Act.

By completing and submitting this questionnaire, YOUR FREE AND INFORMED CONSENT IS IMPLIED and indicates that you understand the conditions of participation in this study. Please find our Letter of Information for Informed Consent here, it includes the details on anonymity, confidentiality, and related issues: goo.gl/EtjT8D

Additional information is also available through our ethics application, protocol number 17-055 from the University of Victoria.

For any questions, please email the research group at courtneywilliams@uvic.ca

Thanks a lot for participating!

A.2 Background information

We would like to gather some background information about you as a researcher.

- How many years have you been conducting software engineering research? [Answer must be a number]
- In what country have you done most of your research? [Short answer]

A.3 Questions about your ICSE paper

- Your ICSE paper title is: [Pre-filled for participant with their ICSE paper title]
- When this research was conducted, you were affiliated with:
  - A university (as faculty, post-doctorate, etc.)
  - A university (as a student)
  - An industrial research laboratory (e.g., Microsoft Research, Google Research Europe, etc.)
  - Other... [short answer]
- Please refer to these definitions of research strategies in the questions below. [See Figure A.1]
- Please select all the research strategies that describe the research conducted in your paper. [Checkboxes]

  - Formal Theory - e.g. theory formulation studies, literature reviews, qualitative synthesis studies, etc.
  - Computational Study - e.g. data mining studies, testing a tool using code repositories or software artifacts, analysis of code, a simulation, etc.
  - Field Study - e.g. observational studies of software development, interview studies in a company, case studies in industry, etc.
  - Field Experiment - e.g. observing participants as they incorporate a tool into their development process, changing the physical layout of a development office and observing its effects, etc.
  - Experimental Simulation - e.g. replicating a participant's normal working conditions in a lab and manipulating some variables during a task, etc.
  - Laboratory Experiment - e.g. providing bug fixing tasks to developers with differing education levels in a controlled, generic laboratory setting, etc.
  - Judgment Study - e.g. sending a prototype tool to developers and asking them to report on which features are incompatible with their development environment, etc.
  - Sample Survey - e.g. an online questionnaire about the challenges developers have understanding debugging tasks, an interview study of how contributors to an open-source project decided to contribute to the project, this study, etc.
  - Other... [short answer]

- Please elaborate on why you used those strategies. *This refers to your answers on the question above.* [Long answer]
- Were there any other research methods used as part of the study but NOT reported in the paper? If yes, please elaborate which ones and WHY they were not described in the paper. *If the answer is no, please enter 'no'.* [Long answer]
- Please refer to these definitions of data sources in the questions below. [See Figure A.2]
- Which data source(s) were used in the research reported in your paper? Please select all that apply. [Checkboxes]

  - Self-Reports - e.g. interviews, focus groups, online surveys, etc.
  - Visible Observer - e.g. coding experiment in a lab, observational case study in industry, etc.
  - Hidden Observer - e.g. observing an online chat for a research team, observing employees who don't know they are being observed, etc.
  - Public Archival Records - e.g. university graduation statistics, hiring statistics, etc.
  - Private Archival Records - e.g. annual performance reviews of a developer, an archive of software design meeting minutes, etc.
  - Trace Measures - e.g. code repositories, software bugs, software development notes, etc.
  - Formal/Theoretical - e.g. the proposal of a new theory, algorithm design with mathematical evaluation, etc.
  - Other... [Short answer]

- Please elaborate on why you chose those data sources. *This refers to your answers on the question above.* [Long answer]
- Please refer to the following research criteria when responding to the questions below. [See Figure A.3]
- Please indicate where you feel the research reported in your paper fits within the following criteria: [Multiple choice grid: 1 - Very low, 2 - Low, 3 - Neutral, 4 - High, 5 - Very High]
  - Generalizability
  - Control
  - Realism
- Please explain your selections above. [Long answer]

A.4 General Research Career Questions

The following questions are about your research work in general; we'd like to like to inquire beyond the the scope of your paper.

Note: If you are responding to this survey for more than one ICSE paper, please complete this section only once, and leave it blank every other time you complete the survey.

- A reminder of the research strategy definitions for the question below. [See Figure A.1]
- Considering the last five years, please rate how often you have USED the following research strategies in your work. *This includes all the studies you participated in as a primary investigator or co-investigator* [Multiple choice grid: Never, Rarely, Sometimes, Often, Most of the time]
  - Formal Theory
  - Computational Study
  - Field Study
  - Field Experiment
  - Experimental Simulation
  - Laboratory Experiment
  - Judgment Study
  - Sample Survey
- How do you define triangulation, and how do you use triangulation in your work? [Long answer]
- In your own work, do you PRIORITIZE any criteria (generalizability, control, or realism) over another? Why? Please elaborate. [Long answer]
- Do you PERCEIVE A BIAS in the software engineering research community regarding certain criteria (generalizability, control, or realism) when it comes to publishing work? Please explain.

A.5 Concluding Questions

- If you would like to be informed about our findings, please enter an email below. *This email address will be confidential, and will be removed from the analysis of the results. We will ONLY use this email address to inform you about our findings. If you're not interested, please leave this field blank.* [Short answer]

- Can we contact you to learn more about your responses? *If yes, please leave your email here. If you're not interested, please leave this field blank.* [Short answer]
- Would you like to comment or add anything else? [Long answer]

A.5.1 Confirmation Message

Thank you very much for your participation!

If you have any questions or would like to know more about our research, please email the research team at courtneywilliams@uvic.ca.

Formal Theory - Research that does not involve gathering new empirical data, but focuses on the creation of models and theories based on previously gathered data. In this case, the primary tool of the researcher is their own mind, manually analyzing and theorizing about existing information.

Computational Study - Controlled computer experiments, computer simulations, and computerized investigations that don't involve any active human participation. The primary tool of the researcher is a computer, and the study or experiment can be conducted as a closed system using previously collected data.

Field Study - The researcher enters a natural setting and conducts research on human participants without manipulating the setting or behavior. This can include, but is not limited to, observing, participating in natural behavior, and asking participants about their behavior.

Field Experiment - The researcher enters a natural setting and observes natural human behavior within that environment having introduced some form of control over one or more variables.

Experimental Simulation - The researcher replicates some aspect of a human participant's natural environment in a controlled experiment. This simulated environment mimics the specific natural setting in which the behavior under study would normally occur as closely as possible. This could include using actors to simulate interactions with others.

Laboratory Experiment – The researcher conducts a controlled experiment with human participants in a laboratory setting without attempting to simulate a natural setting. In this case, the setting in which the behavior occurs is meant to be as generic as possible.

Judgment Study - The researcher canvasses responses from a population with the goal of understanding a phenomenon of interest. However, the researcher is not concerned with understanding a participant's perception or behavior, and is instead using the judgments of participants to gain information about the topic under study itself. The study is constructed in such a way that the responses should be as unaffected as possible by the participant's surroundings.

Sample Survey - The researcher canvasses responses from a population with the goal of understanding human behavior or perception given a phenomenon of interest. The phenomenon itself is not the interest of the study, but rather the human behavior or perception that it causes. The study is constructed in such a way that the responses should be as unaffected as possible by the participant's surroundings.

Figure A.1: The research strategy descriptions provided to participants

Self-Reports - Participants voluntarily report on their own behavior or perceptions for research purposes.

Visible Observer - Observations of human participants who are aware they are being observed for research purposes. This does not necessarily mean that the participant can see the researcher; they could simply be aware that they are being recorded by video or being watched through a one-way mirror.

Hidden Observer - Observations of human participants who are unaware they are being observed for research purposes. The behavior being observed is occurring in real time; this does not include records of behavior that occurred prior to the start of the research.

Public Archival Records - Collecting records of human behavior that were recorded by a third party for non-research purposes that are considered public information. For example, demographic information about publishers at a research conference is recorded by conference organizers and is released publicly, but is not recorded specifically for the purposes of research.

Private Archival Records - Collecting records of human behavior recorded by a third party for non-research purposes that are not meant to become publicly available. For example, details about someone's employment history at a software development company is not meant to become public or to be used for research, and is recorded by a third party (the company).

Trace Measures - Collecting records indirectly created by humans because of their behavior. Participants create these measures on their own. They are not recorded by a third party and they are not created for the purposes of research. Most software development artifacts fall into this category as they are *traces* created because of their software development behavior.

Formal/Theoretical - A study with no human involvement other than the researchers themselves.

Figure A.2: The data source descriptions provided to participants

Generalizability – The extent to which the findings of a study are applicable to the population outside the specific human actors under study.

Control – The level of control of the measurement of behaviors under study, as well as any extraneous factors not under study. A study high in control means that it is unlikely that the human behaviors measured in the study were due to extraneous factors not considered in the study.

Realism – How closely the context under which evidence is gathered matches real life scenarios.

Figure A.3: The desirable research criteria descriptions provided to participants

Appendix B

Ethics Documents

B.1 Implied Consent Form

Method Choice in the Software Engineering Research Community

You are invited to participate in a study entitled Method Choice in the Software Engineering Research Community that is being conducted by Courtney Bornholdt.

Courtney Bornholdt is a graduate student in the Computer Human Interaction and Software Engineering Lab (CHISEL group) in the department of Computer Science at the University of Victoria and you may contact her if you have further questions by [Redacted].

As a graduate student, I am required to conduct research as part of the requirements for a degree in Computer Science. It is being conducted under the supervision of Dr. Margaret-Anne Storey. You may contact my supervisor at [Redacted].

My co-investigators will be Alexey Zagalsky, a PhD student in the CHISEL group, and Dr. Eirini Kalliamvakou, a post-doctoral researcher in the CHISEL group. You may contact them at [Redacted] and [Redacted], respectively.

Purpose and Objectives

The purpose of this research project is to gain insights into the perception of authors in the software engineering research community about their own published work, why they choose certain research strategies and data collection methods, and they perceptions about the software engineering research community.

Importance of this Research

This research is important because by gaining a better understanding of these issues, we can begin to propose ideas and solutions that will increase the overall diversity and quality of our research output as a community.

Participants Selection

You are being asked to participate in this study because you have recently published a paper in a prominent software engineering research venue.

What is involved

If you consent to voluntarily participate in this research, your participation will include the completion of an online survey using Google Forms. The survey should take approximately 15 minutes to complete.

Please be advised that information about you that is gathered for this research study, including information that identifies you as one of the authors of your paper, uses an online program located in the U.S. or a program that can be accessed from the US (Google Forms, GitHub). As such, there is a possibility that information about you may be accessed without your knowledge or consent by the US government in compliance with the US Freedom Act.

Inconvenience

Participation in this study may cause some inconvenience to you, including taking time from your day to participate in the interview.

Risks

There are some potential risks to you by participating in this research, including the possibility that you could be identified as a participant in the study through the publicly available knowledge that you published this paper. To mitigate this risk, your data will be anonymized and stored on a private GitHub repository. All published information will be aggregated with other participant data to mitigate the risk of others identifying your responses.

Benefits

The potential benefits of your participation in this research include increased awareness about important factors in research method choice in software engineering research.

Voluntary Participation

Your participation in this research must be completely voluntary. If you do decide to participate, you may withdraw at any time without any explanation. If you do withdraw from the study your data will be destroyed and will not be included in any analysis.

Anonymity

In terms of protecting your anonymity, we will anonymize your responses before storing them on a private GitHub repository. Any published findings will be aggregated with other participant responses to further anonymize your data.

Confidentiality

Your confidentiality and the confidentiality of the data will be protected by anonymizing your data before storage. Please be advised that this research study includes data storage in the U.S.A. As such, there is a possibility that information about you that is gathered for this research study may be accessed without your knowledge or consent by the U.S. government in compliance with the U.S. Patriot Act, even if it is anonymized. Any data with identifiers will be stored on a password protect University of Victoria computer.

Dissemination of Results

It is anticipated that the results of this study will be shared with others through a paper that will be submitted for publication in research conferences which publish accepted papers publicly online. We also plan to make the results of the study publicly available through a research blog post. It may also be used as part of Courtney Bornholdts masters thesis.

Disposal of Data

Data from this study will be disposed of by deleting our GitHub repository and any locally-stored electronic files.

Contacts

Individuals that may be contacted regarding this study include Courtney Bornholdt, Alexey Zagalsky, Dr. Eirini Kalliamvakou, and Dr. Margaret-Anne Storey, whose contact information is at the beginning of this consent form.

In addition, you may verify the ethical approval of this study, or raise any concerns you might have, by contacting the Human Research Ethics Office at the University of Victoria (250-472-4545 or ethics@uvic.ca).

By completing and submitting the questionnaire, **YOUR FREE AND INFORMED CONSENT IS IMPLIED** and indicates that you understand the above conditions of participation in this study and that you have had the opportunity to have your questions answered by the researchers.

Please retain a copy of this letter for your reference.

B.2 Email Invitation Script

Subject: Help us understand research method choices in the Software Engineering Research Community

Dear *Participant Name*,

We are Courtney Bornholdt, Alexey Zagalsky, Eirini Kalliamvakou, and Margaret-Anne Storey, researchers from the Computer Human Interaction and Software Engineering Lab (CHISEL) in the Department of Computer Science at the University of Victoria, writing to request your participation in a research project.

As an author of *PAPER TITLE*, we would be grateful if you could help us to understand your research method choices and perceptions about software engineering research by completing an online survey. The survey should take about 15-20 minutes. We will openly publish the results so everyone can benefit from them.

Survey on Method Choice in the Software Engineering Research Community *(Link embedded)*

All data will be stored securely. We will openly publish the results so everyone can benefit from them, but we will anonymize everything before doing so. Please note that you are not obligated to participate in the survey or respond to this email. If at some point during the survey you want to stop, you are free to do so without any negative consequences. Please review details of the ethics protocol for this research **goo.gl/EtjT8D**. By filling in and submitting this survey you are providing your implied consent to participate in the study.

We appreciate your participation.

Kind regards,
Courtney Bornholdt

B.3 Reminder Script

Subject: How do SE researchers choose research methods? Help us by participating in our survey Dear *Participant Name*,

Im following up on an earlier invitation to participate in a survey on research methods in software engineering. We are currently soliciting responses until April 17th, and we would really love to have your work represented in our analysis. As an author of *PAPER TITLE*, your opinions and experiences are very important to us and would enrich our study. If you would like to participate, the link to our survey is:

Survey on Method Choice in the Software Engineering Research Community *(Link embedded)*

Please note that participation in the survey is voluntary.

Thank you for your time,
Courtney Bornholdt

B.4 Member Checking Form Letter

Hello *Participant Name*,

I wanted to thank you for participating in my survey, Method Choice in Software Engineering. Your responses were very informative and helped me to create a rich and insightful analysis into why we choose certain research methods in software engineering and our priorities as researchers.

To ensure that our research is as rigorous as possible, I am hoping that you would be willing to clarify some details about your ICSE paper, *ICSE paper title*.

In the survey, you indicated that *Participant's description of their work from the survey.* It is my understanding that *my interpretation of their work.*

Would you say that this is an accurate description of the research in your paper?

Thank you for your time,
Courtney

Appendix C

Coding Process Diaries

This appendix presents the diaries I kept during the process of coding, to show transparency into the how decisions were made in coding.

C.1 RQ2 Analysis Diary

C.1.1 Round 1

Dates: May 9 - 10, 2018

Created and tagged the first codes, see Figure C.1.

C.1.2 Round 2

Date: May 11th, 2018.

Changes:

- Merged "logistical issues" into "because of a particular method" (better fit)
- Renamed "because they are common/accepted by the community" to "used by others/standard practice"
- Merged "because they are efficient" with "ease of use" into "efficiency/ease of use"
- Merged "to compare against existing benchmarks/tools" with "to evaluate efficiency/effectiveness of an approach" into "evaluating an approac"

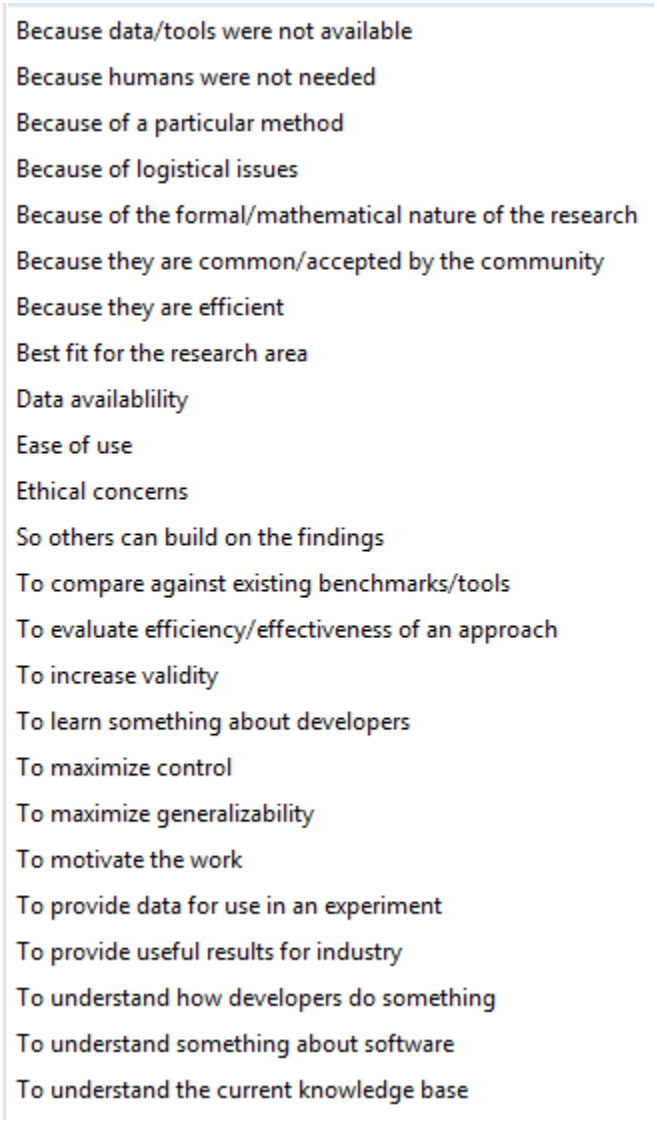


Figure C.1: RQ2 analysis codes after round one.

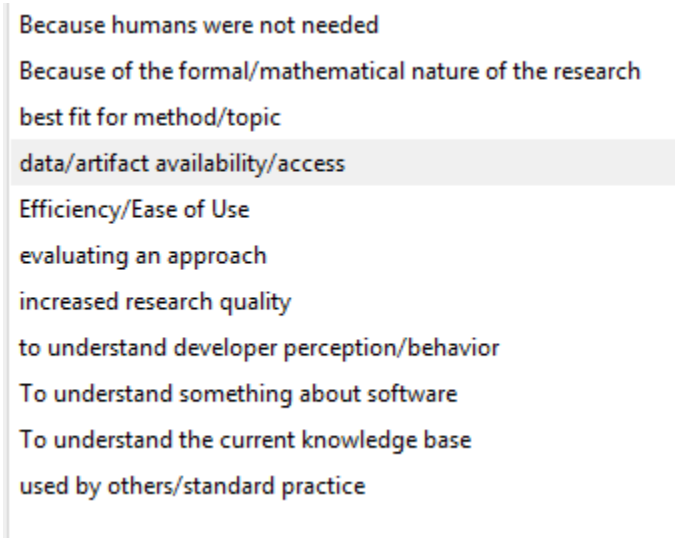


Figure C.2: Final RQ2 analysis codes, after round 2 of coding.

- Merged "to learn something about developers" and "to understand how developers do something" into "to understand developer perception/behavior"
- Renamed "to increase validity" to "increased research quality" and merged in "to maximize control" and "to maximize generalizability"
- Merged "to motivate the work" into "evaluating an approach" (since thats more accurate)
- Merged "to provide useful results for industry" with "so others can build on the findings" into "increased research quality"
- Merged "ethical concerns" into "data availability"
- Merged "best fit for the research area" and "because of a particular method" into "best fit for method/topic"
- Merged "because data/tools were not available" into "data availability"
- Renamed "data availability" into "data/artifact availability/access"
- Merged "to provide data for use in an experiment" into "data/artifact availability/access"

For final code set, see Figure C.2.

C.1.3 Code Analysis and Interpretation

May 14th, 2018: Started writeup

May 15th, 2018: Finished writeup, wrote interpretation

C.2 RQ3 Analysis Diary

C.2.1 Round 1

Date: April 26th, 2018

Created and tagged the first codes, see Figure C.3.

C.2.2 Round 2

Date: May 1, 2018

Changes:

- "Triangulation Irrelevant" merged into "Triangulation Practices"
- "Tools/techniques": Only one text, belongs in "Realism Prioritized in SERC"
- "Control misconceptions" deleted (no text)
- "Data convenience" renamed to "Access to Data"
- "Generalizability Misconceptions" renamed to "Generalizability Problems"
- "Not using triangulation" merged into "Triangulation practices"
- "Problems with Generalizability" merged into "Generalizability Problems"
- "Realism problems" merged with "realism misconceptions"
- "Topic area choice restriction" deleted (no text)

Created code categories:

- Community priorities
  - Control not prioritized in SERC
  - Control prioritized in SERC

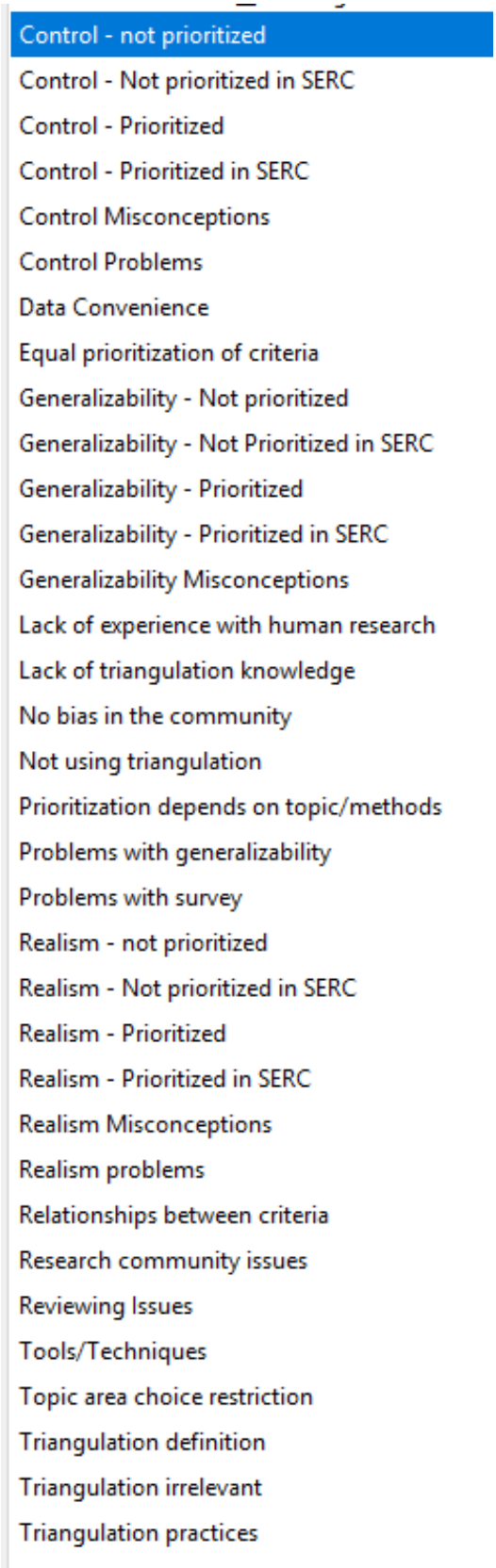


Figure C.3: RQ3 analysis codes after round one.

  - Generalizability not prioritized in SERC
  - Generalizability prioritized in SERC
  - Realism not prioritized in SERC
  - Realism prioritized in SERC
  - No bias in the community
- Personal priorities
  - Control prioritized
  - Control not prioritized
  - Generalizability prioritized
  - Generalizability not prioritized
  - Realism prioritized
  - Realism not prioritized
  - Prioritization depends on topic/methods
  - Equal prioritization of criteria
- Criteria issues
  - Control problems
  - Generalizability problems
  - Realism problems
  - Relationships between criteria
- Systemic Issues
  - Access to Data
  - Lack of experience with human research
  - Research community issues
  - Reviewing issues
- Triangulation
  - Lack of triangulation knowledge
  - Triangulation definition
  - Triangulation practices

C.2.3 Round 3

Date: May 2nd, 2018

Going back over the codes to pull out motivations and reasons for different behaviors and preferences. Codes created are:

- Acceptance by others
- Applicability in real life
- Deeming everything equal
- Difficulty
- For accuracy
- For long term impact
- For reliability
- Funding
- Impact on other criteria
- Increase validity/rigor
- Lack of understanding
- Not important
- To fit the method/topic
- To increase the likelihood of publication
- To increase understanding
- To meet the needs of industrial partners
- To support another criteria

After refining the above codes, I made a series of changes:

- Lumped "for accuracy" into "increase validity/rigor"
- Renamed "increase validity/rigor" into "to increase aspects of research quality"

- Lumped “for long term impact” and “for reliability” into “to increase aspects of research quality”
- Deleted “to support another criteria” only one text, which is covered by other codes
- Renamed “acceptance by others” to “To make it interesting” and moved one code (P14) to “to increase aspects of research quality”
- Removed “funding” Only two participants, one was already covered by another code.
- Removed “not important” Only one participant, what they mean is unclear
- Removed “to increase understanding” two of the codes relate to understanding triangulation, the other way moved to “to increase aspects of research quality”
- Removed “lack of understanding” Doesnt show personal motivations, everything is covered by other codes
- Removed “impact on other criteria” only two quotes which are well-covered by other codes
- Merged “applicability in real life” and “to meet the needs of industrial partners” into “for applicability in software development”
- Removed “to make it interesting” There were only two participants involved, and only one participant seemed to think it was important in any way, the other mentioned it in a minor way

Created the code category “Reasons” which includes:

- Access to Data
- Deeming everything equal
- Difficulty
- To increase aspects of research quality
- Best fit for method/topic
- To increase the likelihood of publication
- For applicability in software development

C.2.4 Code Analysis and Interpretation

3 May, 2018: Wrote up codes for the "Reasons" category

- For each code, collected and explained the content in prose, and summarizing the description of the code at the end in one to two sentences.

- Collected all summaries for the code, and connected the content of the codes, drawing interpretations from the code category.

4 May, 2018: Wrote up codes for the "Triangulation" category

7 May, 2018: Wrote up codes for the category "Personal Priorities" and "Community Priorities"

8 May, 2018: Prepared validation package for Eirini, using P1-P10 and preparing a description of each of the codes.

15 May, 2018: Wrote up codes for the "Criteria Issues" category

Appendix D

Coding Replication Package

The following appendix is the coding replication package that I provided to my second coder, Eirini, for the RQ3 long-answer question analysis.

D.1 Coding Instructions

All of the participant text files are arranged as such:

- Participant #
- How long they have been conducting SE research
- The country in which they did most of their research
- Their ICSE paper title
- Their employment affiliation at the time of the research
- Their answer to: How do you define triangulation, and how do you use triangulation in your work?
- Their answer to: In your own work, do you PRIORITIZE any criteria (generalizability, control, or realism) over another? Why? Please elaborate.
- Their answer to: Do you PERCEIVE A BIAS in the software engineering research community regarding certain criteria (generalizability, control, or realism) when it comes to publishing work? Please explain.
- Would you like to comment or add anything else?

Not all participants answered all questions, so there may be blanks or empty lines indicating no answer. There is one other code, "survey issues", that was used to denote any possible limitations of the survey itself. All of the other codes (and their categories) are below. Note, each category is independent of the others, and there may be significant overlapping codes.

D.2 Communitiy Priorities

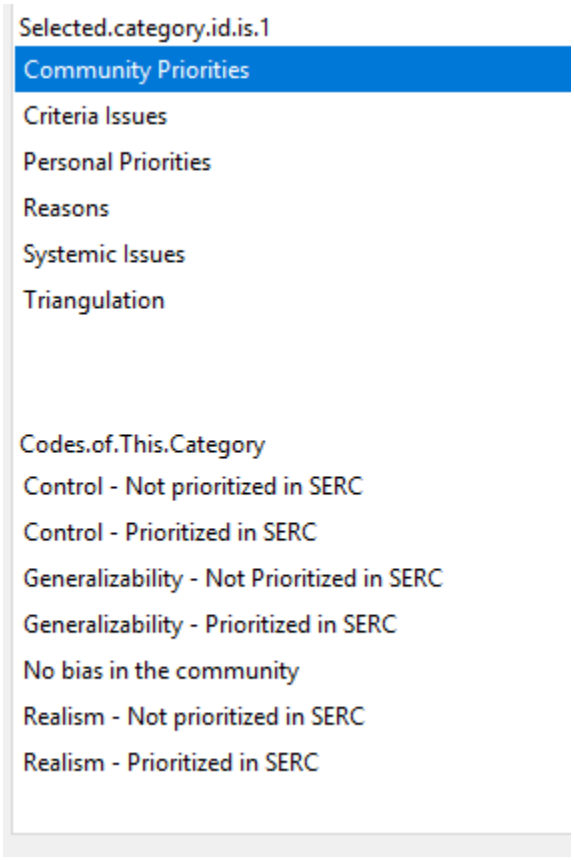


Figure D.1: Codes in the Community Priorities category.

This code category is for when participants discuss community biases that they perceive. For each criterion, there is a code for when the participant says that criterion is under-used or there is a bias against it, that is "Criterion not prioritized in SERC". When they say that it is over-rated or there is a bias in favor, that is "Criterion prioritized in SERC". There is also a code for participants who said there wasnt a bias

in the community. For each code, also include reasoning and context to understand why they think there is/isnt a bias.

D.3 Criteria Issues

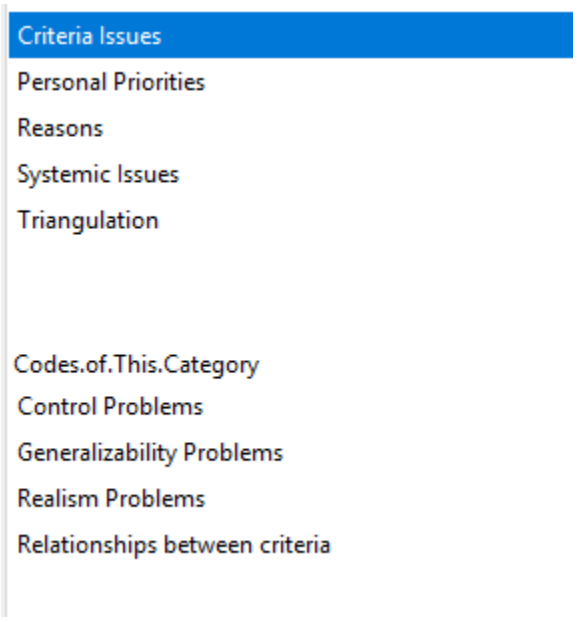


Figure D.2: Codes in the Criteria Issues category.

This category is meant to address the relationships between the criteria and problems that some participants discussed with regards to the ability of researchers to maximize them, etc.

D.4 Personal Priorities

This category discusses whether or not participants prioritize any criteria in their own research or not. Like before, each criteria has its own code for both prioritization, and choosing not to prioritize it. The "not prioritized" code is for criteria that is last on their list; if they prioritize criteria A, then B, and then C last, then code criteria A and B as prioritized and C as not prioritized. Include all context necessary to understand their reasoning. Equal prioritization is reserved for when they say they do not prioritize and view the criteria equally, and prioritization depends on

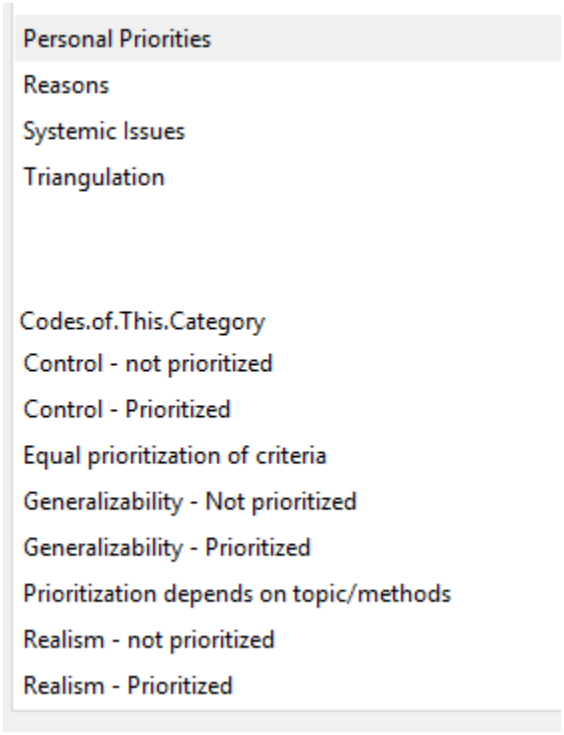


Figure D.3: Codes in the Personal Priorities category.

topic/methods is used to refer to a participant who says that their priorities depend on their topic area or the methods that theyve chosen to use.

D.5 Reasons

This category refers to a participants personal reasoning for prioritizing a certain criteria or making some other choice about their research.

- Access to Data: They make a certain choice because they have access to, or dont have access to, some type of data
- Best fit for method/topic: They choose to prioritize based on what they think best fits their topic or method choice
- Deeming everything equal: They make certain choices out of a belief that all of the criteria are equally valid

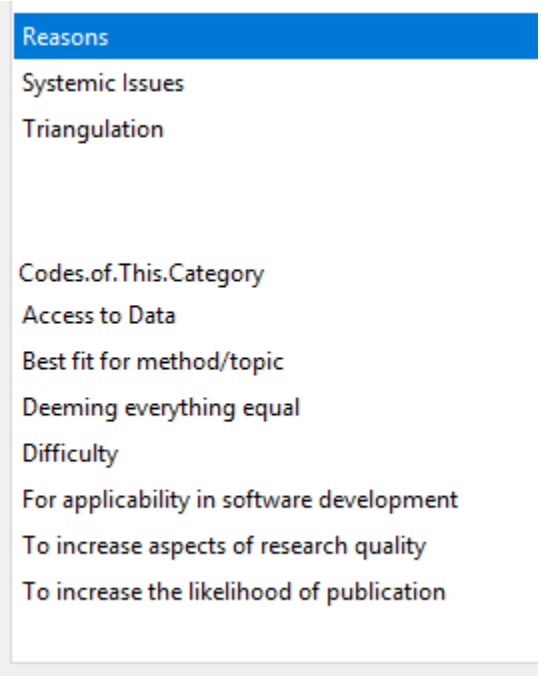


Figure D.4: Codes in the Reasons category.

- Difficulty: They make certain choices because of the difficulty level of something. This could be because doing something would be easy, or because doing something would be very difficult.
- For applicability in software development: They make choices so that their research will apply well to real-life software development contexts and have impact in that field
- To increase aspects of research quality: They make certain choices to maximize some other aspect of research quality beyond the three criteria under the scope of our research
- To increase the likelihood of publication: To make it more likely that their work will be accepted by reviewers and published

D.6 Systemic Issues

This category is used to discuss issues within the software engineering research domain that are beyond prioritizations and biases around the three criteria. Access to data

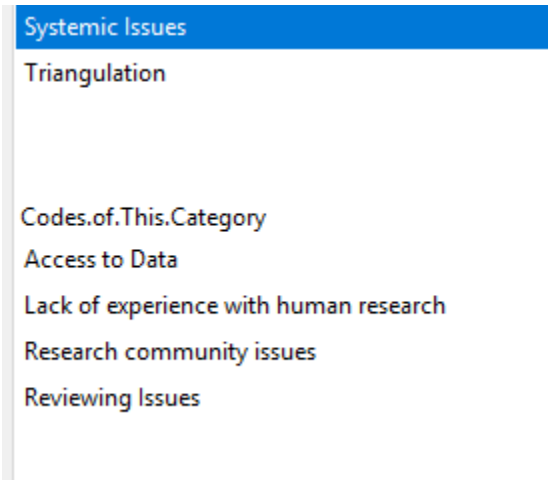


Figure D.5: Codes in the Systemic Issues category.

refers to issues participants may have getting access to data and participants for their work. Lack of experience with human research is exactly that, when a participant says they dont have experience with or dont understand human research. Research community issues is a generic code for issues within the community that dont fall under these other codes, and reviewing issues refers to problems with the review process. Note; there may be overlap between this category and other code categories. For example, if a participant is saying that reviewers prioritize criterion B too much, then that paragraph would be coded both with "Reviewing Issues" and "Criterion B Prioritized in SERC".

D.7 Triangulation

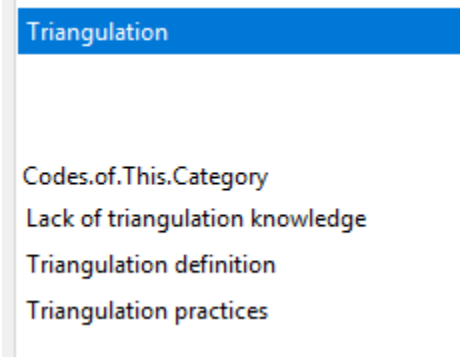


Figure D.6: Codes in the Triangulation category.

This code category is for all things triangulation; lack of triangulation is used to code participants that indicated that they didnt really know what triangulation meant in SE. Triangulation definition is used for any descriptions that participants gave to define triangulation, and triangulation practices is used to code any descriptions that participants might have given of what they do to implement triangulation in their own work.

Appendix E

Synthesis of the Codes

The following appendix presents one of the "writeups" for RQ3, the document that I used to detail my process of synthesizing and interpreting the different code categories.

E.1 Code Category: Reasons

E.1.1 Difficulty

When participants talk about prioritizing certain criteria, one thing they tend to mention is the difficulty levels associated with maximizing them. Generalizability emerged as the most difficult criteria to maximize in research because software companies and practices can vary wildly, making it difficult to generalize the utility of a new tool or technique to all software developers. For example, participants mentioned that high generalizability is "often prohibitive due to the number of datapoints that would be needed and a high realism often makes the number of variables involved unmanageable" (P59). This makes sense, as overwhelmingly participants pointed to a prioritization of realism, so if realism is being maximized it would be difficult to simultaneously maximize generalizability.

One participant pointed to epistemology as an issue for case studies, saying "[methodology] are not a type of study with a positivist view, therefore the results are commonly context dependent" (P30), which points to the potential for other research grounded in more constructivist worldviews to have similar issues with generalizability. They also discuss reviewer expectations with regards to sample size and generalizability, saying that "some reviewers do not understand how challenging it is to obtain real-world data and expect unrealistic amounts of it" (P10). Participants

also mention the difficulties associated with trying to generalize solution-focused research, saying "it is not realistic to come up with a universal solution that works in every context" (P1) and "generalizability comes last since I am completely aware of the fact that real-world settings may significantly vary between different companies" (P10). One participant actually mentioned generalizability being easier, saying "generalizability seems easier to tackle in more formal/theoretical research" (P32), can be explained with the circumplex model because this is where generalizability is at its maximum.

Participants also mentioned problems exerting control in their research. The same as above, with participants tending to prioritize realism, one participant mentioned that "because software engineering is an applied field, it is difficult to have control and still have a realistic experiment that researchers in SE communities are excited about" (P3). One said it is "the hardest one to get right" (P7), which may explain another participants view that "running experiments in a near-real environment with real professional developers are much harder for a graduate student" (P44).

No participants said they experienced difficulties prioritizing realism in their work.

Quotes

- "high generalizability is often prohibitive due to the number of datapoints that would be needed and a high realism often makes the number of variables involved unmanageable." (P59)
- "Generalization is quite difficult to say because usually, [methodology] are not a type of study with a positivist view, therefore the results are commonly context dependent." (P30)
- "It seems that some reviewers do not understand how challenging it is to obtain real-world data and expect unrealistic amounts of it." (P10)
- "In my field of work, it is not realistic to come up with a universal solution that works in every context." (P1)
- "Generalizability comes last since I am completely aware of the fact that the real-world settings may significantly vary between different companies." (P10)
- "I value all three aspects; yet generalizability seems easier to tackle in more formal/theoretical research." (P32)

- "Because software engineering is an applied field, it is difficult to have control and still have a realistic experiment that researchers in SE communities are excited about. So we often give up control for realism." (P3)
- "I would prefer control, as it is the hardest one to get right" (P7)
- "'Control' over others, because it's usually under our control. Running experiments in a near-real environment with real professional developers are much harder for a grad student." (P34)

Summary

Generalizability is very difficult to achieve in SE for reasons like industry variation and epistemology, and reviewers have unrealistic expectations of sample size. Control is also considered difficult to achieve, while realism was not considered difficult.

E.1.2 Access to Data

Some participants mentioned issues with access to data. One saying that they didnt conduct a field experiment in their work because "it is too expensive regarding controlling experimental variables and hiring the participants" (P39), and another saying that "having no access to the actual engineers or processes is a huge problem and limits the researchers ability to include reality and natural settings" (P31) in their work. This difficulty accessing data may be prohibiting researchers from making some choices that might make their research more impactful, and one participant even commented that "just because data is convenient or available does not mean it reveals what we are looking for" (P4), suggesting that while there are difficulties associated with some methods or data sources, choosing the easier route or more publicly available data may introduce issues with construct validity.

Summary

It is difficult to get access to realistic data, and this may lead to choosing readily available data that is not as useful and leads to issues with construct validity.

E.1.3 For applicability in software development

When discussing their motivations for making certain choices, a dominant factor was making sure that it would be usable in real-life software development. Some participants were mostly concerned with the impact they could have in broad software development environments, saying they "strive for more applicable solutions that could be adopted by developers in the future" (P1), that they aim to "solve a problem that is commonly encountered by software developers" (P22), they need to have "practical impact" (P43) and their work needs to "have relevance for both practitioners and researchers in the discipline" (P6).

Some participants said they needed their work to be high in realism because of their involvement with industry; One participant even said that because they are employed in industry everything "all my research needs to help practitioners in the real-world daily work" (P10). This goes both ways, with one participant commenting "to prioritize realism I will need much closer cooperation with industry" (P42), suggesting a strong relationship between industry collaborations and highly realistic research. This is supported by our research lens, where the field strategies and computational studies using trace measures of industry software development are high in realism.

Most participants who mentioned industry were in the form of collaborations, and needing to satisfy the needs of their industry participants to continue the relationships and to satisfy issues with research funding. In these situations, participants mentioned that "interests of the partners, imposes us to be more realist, a bit less generalist, and to handle control as best as we can" (P25). This view was shared by other participants who said that while realism had the highest priority, generalizing their solution to as many different software development contexts as possible was the next priority. One participant said "it is also important to make sure that the approach could generalize for subjects beyond the current study to ensure the applicability of the approach" (P22), and another saying that generalizability comes after realism because "it is very important to select subject software systems, which are representative of the practices applied in software engineering" (P34).

Because SE research is typically solution-focused towards a problem that exists in industry, this makes sense. First, you want to make sure you provide a solution for at least someone, particularly the person who is participating in the research with you. Then, where possible, you try to make sure that your solution will help not only the

team who is working with you, but be general enough to help others. After trying to maximize these two criteria, it would be incredibly difficult to address control in any way without conducting complementary studies.

Summary

In order to have real world impact and satisfy industry participants, the order of priorities is realism ¿ generalizability ¿ control

E.1.4 Best fit for method/topic

Many participants discussed their priorities changing based on the topic or method they are using in their research. Participants who focus on experiments, particularly lab experiments say they "aim for precision of measurement" (P46) and "need to control everything as much as possible to reduce noise" (43), but that this prioritization of control "makes the results of my experiments more reliable" (P42) and "if you cannot control the variables, the experiment result is meaningless" (P39). Other participants focused on realism because of their topic or methods, saying that "realism is #1 in [topic]" (P36), "realism is the point of a case study" (P12), and "realism always comes first because most of my research operates on real programs" (P5).

Most participants who talked about matching their priorities talked about it in more general terms, saying their priorities "would depend heavily on what the aim of the study was" (P48) and that they try to balance all of them "compatibly with the specificities of the domain under investigation" (P41). This is supports our research lens, where some research strategies make it ideal to prioritize one criteria over another simply because of the nature of the methods.

Summary

Participants prioritize criteria differently depending on the methods theyre using, and they prioritize the criteria that is the best fit for the method (the closest criteria on the circumplex)

E.1.5 To increase aspects of research quality

Some participants mentioned prioritizing certain criteria because of certain benefits that prioritization would provide them in their work, increasing its overall quality.

Researchers mentioning prioritizing generalizability said that it "helps us understand the relevance of an approach" (P38) and have "impact in the long term" (P42), with one participant even saying that "if an approach is not generalizable, it makes no sense to others" (P14).

Participants who maximized control said that it helps you to "gain an accurate picture of your research" (P45), that it allows for "fair evaluation of the proposed approach" (P22), and allows them to "find differences between very specific variables" (P15). Realism was not specifically mentioned here, but this is covered by the code "for applicability in software development", where many participants discussed increased impact from highly realistic research.

Summary

Some participants prioritize generalizability or control because of other benefits for overall research quality.

E.1.6 Deeming everything equal

Many participants recognized the utility of all of the criteria equally, saying that they try not to prioritize because they are all important. They "deem all of them equally important" (P2) and "value all three aspects" (P32), but many recognize that the ability to do this "depends on the context of the research" (P25) in reality. In addition to the already-discussed reason that prioritization depends on the method or topic, one participant also mentioned that "getting all three is ideal, but this may be better-suited to multiple smaller projects as opposed to one big one" (P16). This is a critical insight, and shows that at least one participant is aware of the utility of triangulating with multiple strategies to achieve high levels of all three criteria.

Summary

Some participants value all three criteria equally, while accepting some limitations that make this unrealistic for a single study.

E.1.7 To increase the likelihood of publication

A big reason for many participants to prioritize certain criteria is to increase their likelihood of publication. Overwhelmingly, participants emphasized the importance of

high generalizability for publication. They said thats “what reviewers easily criticize” (P19), making it “difficult to publish work thats not generalizable” (P18).” As a symptom of this, participants said that “this is why there is an increasing number of subjects” (P34) in studies, and a “more is better” (60) attitude in the community. This can be harmful for naturalistic studies, and as previously mentioned, participants talk about how difficult it is to obtain the amount of data necessary to conduct a study that maximizes generalizability. Only one participant mentioned prioritizing realism or control for the purposes of publication, saying that “without realism it is hard to make the case that the research is useful, and without control, it is not clear where it is applied” (P33).

Summary

Generalizability is often prioritized for the purposes of publication because it is easily criticized by reviewers who often have unrealistic expectations of sample size.

E.1.8 Summaries

- Generalizability is very difficult to achieve in SE for reasons like industry variation and epistemology, and reviewers have unrealistic expectations of sample size. Control is also considered difficult to achieve, while realism was not considered difficult.
- It is difficult to get access to realistic data, and this may lead to choosing readily available data that is not as useful and leads to issues with construct validity.
- In order to have real world impact and satisfy industry participants, the order of priorities is realism ¿ generalizability ¿ control
- Participants prioritize criteria differently depending on the methods theyre using, and they prioritize the criteria that is the best fit for the method (the closest criteria on the circumplex)
- Some participants value all three criteria equally, while accepting some limitations that make this unrealistic for a single study.
- Some participants prioritize generalizability or control because of other benefits for overall research quality.

- Generalizability is often prioritized for the purposes of publication because it is easily criticized by reviewers who often have unrealistic expectations of sample size.

E.1.9 Findings and Interpretations

While most participants seem to view all three of the criteria as equal, there are many contextual issues and difficulties that arise that are skewing the community away from equal prioritization of the three criteria. First, software engineering research tends to be solution-focused and involve close partnerships with industry. This means that to maintain these relationships with industry participants, and to come up with solutions that are accepted by the community, there is a high level of prioritization of realism. However, accessing this data and these industry participants can be difficult for some people, leading to issues with construct validity when researchers compromise and prioritize realism, but use easily accessed data that may not accurately represent reality.

Second, there are significant issues surrounding generalizability in software engineering research. After focusing on realism for solution-focused industry research, which represents a large amount of published research, authors are then trying to prioritize generalizing their solutions to apply to as many software development contexts as possible. This is for many reasons, including a desire to have lasting impact in the community, however for many participants it is out of a desire to have their work published. Generalizability is the easiest of the criteria to criticize as a reviewer, and papers are rejected because they are considered not generalizable enough, and even though they are highly realistic, they are not considered to be applicable outside of the company or organization used in the study. Generalizability is incredibly difficult to achieve, considering that there is wide variation in the types of software development organizations that exist in modern software engineering. Between the immense amount of contextual data necessary to understand the workings of a software organization and its needs, and the difficulties associated with accessing data about these organizations, it is incredibly difficult to generalize research in software engineering. With reviewers expecting larger and larger sample sizes, and software development organizations becoming more and more diverse, it is becoming increasingly unrealistic to meet reviewer expectations of generalizability. This is leading some reviewers to

do far more work than is necessary to conduct meaningful and rigorous research just to be published.

Some participants prioritize control, but this tends to be based on a desire to prioritize the criteria that match the research strategy that they are using, which is an attitude held by many participants. Researchers conducting experimental work, especially with human participants, tended to be concerned with control in their research. Researchers who focused on behavioral research and field work were much more concerned overall with realism. This is a sensible approach that is supported by the circumplex, which shows that field studies are naturally high in realism, and the experimental strategies are high in control.

Overall, there are two important takeaways from this analysis. We are incredibly concerned with realism as authors, and this is often to maximize our impact with our industry partners and create solutions that can be used in software development and solve real-life development problems. However, there is a disconnect our priorities as researchers and authors and the priorities of reviewers of our work; it is easy to criticize and reject a study for a lack of generalizability, even though most authors are more concerned with their works applicability to specific software development contexts. This is leading to researchers striving for ever more generalizable results while still maintaining the realism that satisfies industry. This means that increasingly large amounts of effort must go into collecting and analyzing incredibly large samples of data on which to base your findings, even if it is not necessary, or may not even generalize to the diverse field of software development just to achieve publication.