A Meta-Research Agenda for Recruitment and Study Design for Developer Studies

Matthew Smith
University of Bonn, Fraunhofer FKIE
smith@cs.uni-bonn.de

Anastasia Danilova
University of Bonn
danilova@cs.uni-bonn.de

Alena Naiakshina
Ruhr University Bochum
alena.naiakshina@rub.de

ABSTRACT

Human-Computer Interaction (HCI) researchers have been studying end-users for many decades using a rich toolbox of study methodology for their work. Thus ample knowledge exists on how to recruit and conduct studies with end-users. While studying and helping end-users is a worthwhile goal, developers are humans too and studying and helping them is also essential. However, recruiting and studying developers is much more challenging than end-users because we still lack much of the best practice knowledge we have with end-users. In particular, recruitment is a challenge that is affected by both the scarcity of developers, financial constraints but also by the study design. There are still many open questions: Can we recruit CS students as proxies for company developers? Can we recruit online Freelancers to get large sample sizes? How high does the compensation need to be? Can we break down complex tasks into smaller ones to ease recruitment? In this paper, we present a meta-research agenda to set up a framework to answer these questions.

ACM Reference Format:
Matthew Smith, Anastasia Danilova, and Alena Naiakshina. 2022. A Meta-Research Agenda for Recruitment and Study Design for Developer Studies. In *International Workshop on Recruiting Participants for Empirical Software Engineering (RoPES '22), May 17, 2022, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 3 pages.

1 INTRODUCTION

A common approach in software engineering studies is the recruitment of convenience samples, such as computer science (CS) students (e.g., [1, 4, 7, 15, 20, 23, 28]). To widen the recruitment pool and include non-student participants, it is common for researchers to resort to online studies and recruit participants online (e.g., [2, 3, 5, 14, 16, 17, 25, 29, 31]). Diverse recruitment strategies have been used, such as cold-calling programmers on platforms such as Stack Overflow, GitHub, Meet-up groups, etc. or posting open invitations on social media, in forums, newsletters, and events, with the expectation being that participants without programming knowledge will not sign up for the studies [3, 6, 26]. Recruitment of developers and study design are closely intertwined topics. It is considered best practice that end-user studies should not be longer than 20 minutes. However, programming tasks are often complex and take hours or days. This significantly impacts recruitment and compensation. Recruiting developers to spend 20 minutes on an online survey does not pose the same challenge as asking them to spend eight hours in a lab. There are many variables that affect recruitment: Incentives, target developers (e.g., CS students, freelancers, company developers), task design, deception (very relevant for security studies), location and type of study (online, lab, field, qualitative, quantitative). We consider the effect these study design variables have to be a multi-dimensional meta-research problem and propose a research approach to study these variables. As part of the ERC Project Frontiers of Usable Security, we conducted a series of meta-studies explicitly looking at recruitment and design of developers studies. In the following, we present our methodology to encourage others to also gather evidence on how recruitment and study design affect the research itself.

RoPES '22, May 17, 2022, Pittsburgh, PA, USA

2 RESEARCH METHODOLOGY

Our research methodology consists of two parts: a primary study and a meta-study. The primary study consists of some developer-centric research questions such as: does a particular library help developers to store passwords securely, what effect does pair programming have, etc. This primary study will often contain a variable of interest, such as the used library. In most cases, researchers will design a study based on their best judgment and recruiting capabilities - the latter often being the limiting factor, e.g., a lab study using CS students is used because they are available or an online study is run with volunteers from Github to avoid students, but this limits the task size due to the volunteer nature. Our approach consists of taking such a study and varying the study design variables - called meta-variables in the following - and running a randomized control trial on them. Figure 1 shows an example. Here the primary study consists of a randomized control trial concerning password storage and the independent variable is the library used (JSF vs. Spring). For the meta-study, we run the primary study several times varying the recruitment method. In this case, we have two independent meta-variables (lab vs. online) and (students vs. online freelancers vs. company developers). With this approach, we can gather information on how the different recruitment options affect the study results. In the following, we highlight some meta-research goals:

2.1 Meta-Research Goals

2.1.1 Incentives. Research and systemize how incentives influence the ecological validity of developer studies: End-user studies are usually fairly brief (often less than an hour) but nevertheless cover an entire end-user task (such as creating a password). They are often conducted with a large number of unskilled workers or students (often hundreds and sometimes thousands) with fairly low compensation (sometimes in the sub-euro range). In contrast, most developer tasks are more complex and take more time (hours or even days). In addition, developers are unlikely to be incentivized

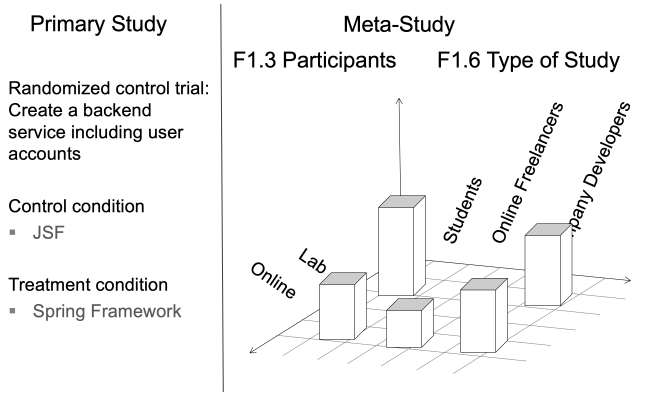

Figure 1: Meta-Research Methodology

to take part in studies in return for a small amount of money. Voluntary participation is uncommon in end-user research, since it is generally believed to bring with it a particularly strong form of self-selection bias, since it mainly recruits altruistic participants. However, this bias might be negligible in the case of developer studies and maybe even be preferable to the skill bias that might be incurred by using financial incentives. Developers tend to lack time rather than money and so financial incentives might not be as effective as altruistic ones. These variables are unknown and need to be researched to enable future developer recruitment to be conducted efficiently while avoiding unnecessary biases created through the participant recruitment method. An example of a study in which we analyzed payment incentives can be found in [17].

2.1.2 Task Design. Research and systemize how task design affects the ecological validity of developer studies: Due to the complexity of development tasks, the effort and time required to complete a task are far greater than for end-user studies. The requirement for a certain level of expertise and the time constraints of most IT security-professionals further increases the difficulty of recruiting large numbers of participants. Thus, it is desirable to be able to split complex tasks into sub-tasks and study them separately. This may, however, significantly affect the study design and potentially the results but is probably a necessity for effective recruitment. An example of a study we conducted can be found in [11].

2.1.3 CS Students, Online Freelancers and Company Developers. Research whether students are a viable proxy for developers in usability studies: Unlike traditional HCI research that discourages the recruitment of students and often completely excludes CS students, CS students might offer a good facsimile for professional developers and thus are a viable population to conveniently conduct studies with. If this is the case, it would greatly ease developer research, since recruiting students to take part in a study tends to be much easier than recruiting professional developers. Online freelancers are another interesting recruitment option since with a sufficient budget, a large number of participants can be recruited very easily. However, without knowing more about the population, it is hard to judge how they compare to students and company developers. An example of a study in which we analyzed different developers can be found in [9, 16, 17, 30].

2.1.4 Deception. Research how deception or lack thereof affects developer usability studies: Deception in studies has a long history of being debated both in terms of ecological validity and ethics [22]. In the Usable Security community, deception is commonly used, since it is believed that if end-users know a study is about their security behavior, then that knowledge will affect the very behavior the researchers are aiming to study [21, 27]. However, this is not true for all studies. We have shown that for an end-user password study we ran, deception had no effect [13], showing the need to study these questions on a case-by-case basis. This also affects recruitment since the study description (with or without deception) could attract different participants. It must also be considered that it might be harder to deceive developers about the true purpose of a study and also annoy them if they agreed to help research a specific topic to then find out it is a different one. Without knowing how this affects recruitment and execution of the study. An example of a study where we analyzed deception can be found in [9, 17–19].

2.1.5 Location and Type. Research the effects the different study forms (lab, online, field) have on developer participants: The study environment can significantly affect end-user behavior. In some studies, participants stated that they consciously engaged in risky behavior that they typically avoided. Their reasoning for this was that they were taking part in a study conducted by a university and they were confident that the researchers would not allow them to be harmed. This effect is even stronger if the experiment is conducted in a lab using the lab's own hardware, since there was a perception that any negative effect would not affect their own property [24, 27]. The work we conducted on the ecological validity of end-user studies has shown significant differences in behavior between online and lab studies, and critically that for certain research questions lab studies are preferable but for others online studies [13]. All these aspects are unknown for developer studies and also significantly impact recruitment. While we have conducted a number of lab, online and field studies, we have not run a direct comparison yet. Work highlighting recruitment risks when conducting online surveys with programmers can be found in [8, 10, 12]

2.1.6 Qualitative vs Quantitative. Finally, we want to encourage the community to consider the effects of qualitative and quantitative research. The same research question can often be studied using both methods, but qualitative studies can often work with smaller sample sizes, thus easing recruitment. An example of a study in which we analyzed deception can be found in [9, 18, 19].

3 CONCLUSION

We have conducted a number of primary and meta-studies to shed light on the effect of study design and recruitment. However, this work can only be seen as a very first step and more diverse work is needed to explore developer study design and recruitment from different angles. We hope that, where possible, researchers conducting a primary study will add a meta-study to help develop our communities' knowledge on study design and improve the scientific quality of our work.

ACKNOWLEDGMENTS

This work was partially funded by the ERC Grant 678341: Frontiers of Usable Security.

REFERENCES

[1] Y. Acar, M. Backes, S. Fahl, D. Kim, M. L. Mazurek, and C. Stransky. 2016. You Get Where You're Looking for: The Impact of Information Sources on Code Security. In *2016 IEEE Symposium on Security and Privacy (SP'16)*. 289–305. https://doi.org/10.1109/SP.2016.25

[2] Yasemin Acar, Christian Stransky, Dominik Wermke, Michelle L Mazurek, and Sascha Fahl. 2017. Security Developer Studies with GitHub Users: Exploring a Convenience Sample. In *Thirteenth Symposium on Usable Privacy and Security (SOUPS'17)*. 81–95.

[3] Hala Assal and Sonia Chiasson. 2019. 'Think Secure from the Beginning': A Survey with Software Developers. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI'19)*. ACM, New York, NY, USA, Article 289, 13 pages. https://doi.org/10.1145/3290605.3300519

[4] Titus Barik, Justin Smith, Kevin Lubick, Elisabeth Holmes, Jing Feng, Emerson Murphy-Hill, and Chris Parnin. 2017. Do Developers Read Compiler Error Messages?. In *Proceedings of the 39th International Conference on Software Engineering* (Buenos Aires, Argentina) *(ICSE '17)*. IEEE Press, Piscataway, NJ, USA, 575–585. https://doi.org/10.1109/ICSE.2017.59

[5] Jason Bau, Frank Wang, Elie Bursztein, Patrick Mutchler, and John C Mitchell. 2012. Vulnerability Factors in New Web Applications: Audit Tools, Developer Selection & Languages. *Stanford, Tech. Rep* (2012).

[6] Moritz Beller, Niels Spruit, Diomidis Spinellis, and Andy Zaidman. 2018. On the Dichotomy of Debugging Behavior Among Programmers. In *Proceedings of the 40th International Conference on Software Engineering (ICSE'18)*. 572–583. https://doi.org/10.1145/3180155.3180175

[7] Patrik Berander. 2004. Using Students as Subjects in Requirements Prioritization. In *Empirical Software Engineering, 2004. ISESE'04. Proceedings. 2004 International Symposium on*. IEEE, IEEE, Redondo Beach, CA, USA, 167–176.

[8] Anastasia Danilova, Stefan Horstmann, Matthew Smith, and Alena Naiakshina. 2022. To appear: Testing Time Limits in Screener Questions for Online Surveys with Programmers. In *2022 IEEE/ACM International Conference on Software Engineering (ICSE)*. IEEE.

[9] Anastasia Danilova, Alena Naiakshina, Johanna Deuter, and Matthew Smith. 2020. Replication: On the Ecological Validity of Online Security Developer Studies: Exploring Deception in a Password-Storage Study with Freelancers. In *Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020)*. 165–183.

[10] Anastasia Danilova, Alena Naiakshina, Stefan Horstmann, and Matthew Smith. 2021. Do you really code? Designing and Evaluating Screening Questions for Online Surveys with Programmers. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. IEEE, 537–548.

[11] Anastasia Danilova, Alena Naiakshina, Anna Rasgauski, and Matthew Smith. 2021. Code Reviewing as Methodology for Online Security Studies with Developers-A Case Study with Freelancers on Password Storage. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*. 397–416.

[12] Anastasia Danilova, Alena Naiakshina, and Matthew Smith. 2020. One Size Does Not Fit All: A Grounded Theory and Online Survey Study of Developer Preferences for Security Warning Types. In *Proceedings of the 42nd International Conference on Software Engineering (ICSE'20)*. https://doi.org/10.1145/3377811.3380387

[13] S Fahl, M Harbach, Y Acar, and Smith M. 2013. On the ecological validity of a password study. In *Symposium on Security and Privacy*.

[14] Peter Leo Gorski, Luigi Lo Iacono, Dominik Wermke, Christian Stransky, Sebastian Möller, Yasemin Acar, and Sascha Fahl. 2018. Developers Deserve Security Warnings, Too: On the Effect of Integrated Security Advice on Cryptographic API Misuse. In *Fourteenth Symposium on Usable Privacy and Security (SOUPS'18)*. 265–281.

[15] Thomas D. LaToza, Gina Venolia, and Robert DeLine. 2006. Maintaining Mental Models: A Study of Developer Work Habits. In *Proceedings of the 28th International Conference on Software Engineering* (Shanghai, China) *(ICSE '06)*. ACM, New York, NY, USA, 492–501. https://doi.org/10.1145/1134285.1134355

[16] Alena Naiakshina, Anastasia Danilova, Eva Gerlitz, and Matthew Smith. 2020. On Conducting Security Developer Studies with CS Students: Examining a Password-Storage Study with CS Students, Freelancers, and Company Developers. In *Proceedings of the 2020 Conference on Human Factors in Computing Systems (CHI'20)*. 1–13. https://doi.org/10.1145/3313831.3376791

[17] Alena Naiakshina, Anastasia Danilova, Eva Gerlitz, Emanuel von Zezschwitz, and Matthew Smith. 2019. 'If You Want, I Can Store the Encrypted Password': A Password-Storage Field Study with Freelance Developers. In *Proceedings of the 2019 Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI'19)*. ACM, New York, NY, USA, Article 140, 12 pages. https://doi.org/10.1145/3290605.3300370

[18] Alena Naiakshina, Anastasia Danilova, Christian Tiefenau, Marco Herzog, Sergej Dechand, and Matthew Smith. 2017. Why Do Developers Get Password Storage Wrong?: A Qualitative Usability Study. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (Dallas, Texas, USA) *(CCS'17)*. ACM, New York, NY, USA, 311–328. https://doi.org/10.1145/3133956.3134082

[19] Alena Naiakshina, Anastasia Danilova, Christian Tiefenau, and Matthew Smith. 2018. Deception Task Design in Developer Password Studies: Exploring a Student Sample. In *Fourteenth Symposium on Usable Privacy and Security (SOUPS'18)*. USENIX Association, Baltimore, MD, 297–313. https://www.usenix.org/conference/soups2018/presentation/naiakshina

[20] Duc Cuong Nguyen, Dominik Wermke, Yasemin Acar, Michael Backes, Charles Weir, and Sascha Fahl. 2017. A Stitch in Time: Supporting Android Developers in WritingSecure Code. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (Dallas, Texas, USA) *(CCS '17)*. ACM, New York, NY, USA, 1065–1077. https://doi.org/10.1145/3133956.3133977

[21] T OJagatic, N Johnson, M Jakobsson, and F Menczer. 2007. Social Phishing. In *Communications of the ACM*.

[22] M Orne and C Holland. 1968. On the Ecological Validity of Laboratory Deceptions. In *International Journal of Psychiatry*.

[23] Iflaah Salman, Ayse Tosun Misirli, and Natalia Juristo. 2015. Are students representatives of professionals in software engineering experiments?. In *Proceedings of the 37th International Conference on Software Engineering-Volume 1*. IEEE Press, IEEE, Florence, Italy, 666–676.

[24] A Sotirakopoulos and Beznosov K Hawkey, K. 2011. On the Challenges in Usable Security Lab Studies: Lessons Learned from Replicating a Study on SSL Warnings. In *Symposium on Security and Privacy*.

[25] Leonardo Sousa, Roberto Oliveira, Alessandro Garcia, Jaejoon Lee, Tayana Conte, Willian Oizumi, Rafael de Mello, Adriana Lopes, Natasha Valentim, Edson Oliveira, et al. 2017. How Do Software Developers Identify Design Problems? A Qualitative Analysis. In *Proceedings of the 31st Brazilian Symposium on Software Engineering (SBES'17)*. 54–63. https://doi.org/10.1145/3131151.3131168

[26] Davide Spadini, Gül Çalikli, and Alberto Bacchelli. 2020. Primers or Reminders? The Effects of Existing Review Comments on Code Review. In *Proceedings of the 42nd International Conference on Software Engineering (ICSE'20)*. https://doi.org/10.1145/3377811.3380385

[27] J Sunshine, S Egelman, H Almuhmedi, N Atri, and L Cranor. 2009. Crying wolf: An empirical study of SSL warning effectiveness. In *USENIX Security*.

[28] Mikael Svahnberg, Aybüke Aurum, and Claes Wohlin. 2008. Using Students As Subjects - an Empirical Evaluation. In *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement* (Kaiserslautern, Germany) *(ESEM '08)*. ACM, New York, NY, USA, 288–290. https://doi.org/10.1145/1414004.1414055

[29] Chamila Wijayarathna and Nalin A. G. Arachchilage. 2018. Why Johnny Can't Store Passwords Securely? A Usability Evaluation of Bouncycastle Password Hashing. In *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018* (Christchurch, New Zealand) *(EASE'18)*. Association for Computing Machinery, New York, NY, USA, 205–210. https://doi.org/10.1145/3210459.3210483

[30] Khaled Yakdan, Sergej Dechand, Elmar Gerhards-Padilla, and Matthew Smith. 2016. Helping Johnny to Analyze Malware: A Usability-Optimized Decompiler and Malware Analysis User Study. In *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, IEEE, San Jose, CA, USA, 158–177.

[31] Aiko Yamashita and Leon Moonen. 2013. Do Developers Care about Code Smells? An Exploratory Survey. In *2013 20th Working Conference on Reverse Engineering (WCRE'13)*. IEEE, 242–251. https://doi.org/10.1109/WCRE.2013.6671299