Programmer User Studies: Supporting Tools & Features

Lázaro Costa*, Susana Barbosa†, Jácome Cunha*
lazaro@fe.up.pt, susana.a.barbosa@inesctec.pt, jacome@fe.up.pt
*HASLab/INESC TEC, Faculty of Engineering, University of Porto, Portugal
†INESC TEC, Portugal

***Abstract*—User studies are paramount for advancing science. In particular, the empirical evaluation of programmer-oriented tools is important to validate research ideas and prototypes, as well as production-ready tools. Previous research has collected several tools used by the software engineering and behavioral science communities to design and run studies.**

In this work, we study tools used in software engineering studies and identify their features. Furthermore, we analyze three behavioral science experiment tools to identify design ideas that might be adapted to programmer user studies.

With this work, we present the set of features currently offered by software engineering tools to support researchers in the design and execution of programmer user studies. We also present the characteristics of some tools used in behavioral science experiments to identify design ideas that can be adapted to programmer user studies.

***Index Terms*—Empirical Software Engineering, Empirical Evaluation Tools**

I. Introduction

Several research communities, such as software engineering (SE) or human-centric, have, for many years now, imposed the need for empirical evaluation of proposed techniques, methodologies, and corresponding tools. In particular, user studies are now standard when evaluating prototypes and implementing new approaches. Moreover, empirical software engineering evaluation is also common in many software companies. While these studies are common, they are also hard to design and run [1]. Indeed, they require the execution of the study in specific (controlled) conditions and the capture of all the relevant information for further analysis. However, it is difficult to create the required environment and to centralize all the collected information in a simple package for later analysis [2], [3].

The elaboration of user studies and making all the collected information available is crucial to support research claims, research work, and the evaluation of prototypes and tools. Indeed, a significant amount of scientific research is currently supported by user studies. However, it is not easy to plan, execute, capture, and make available all the relevant information of a user study [4]. Thus, planning, executing, capturing, and making all the captured information available in user studies is now more than ever a technological challenge in many research fields [2] such as software engineering [5]–[7], behavior science [8], [9], or quality of (user) experience [10].

There are SE experiment tools available to aid researchers making their user studies, such as ARRESTT [11], eSEE [12], Experiment Manager [13], ExpDSL [14], [15], Ginger2 [16], K-Alpha [17] or SESE [18]. However, these approaches have limitations in terms of recruitment of participants, designing studies using wizards and guides, reusing study configuration files, and reusing configurable task interfaces and components. In the field of behavioral science, there are some tools (Gorilla [19], jsPsych [20] and LookIt [21]) able to solve these limitations, but they are specific to this domain.

In this research, we conduct a literature review of the experiment tools used in software engineering studies and identify their features. Furthermore, we consider three behavioral science experiment tools to identify their features and design ideas that might be adapted to programmer user studies.

To aid the development of more effective evaluation tools, we define the following research question (RQ):

What tools and corresponding features are currently available for supporting user studies?

To answer this RQ, we collected the tools identified in [22] covering the years from 2002 to 2023 and tried to find additional tools built since 2023. Following, we analyzed each tool and identified its relevant features for user studies (see Section III).

Based on the answer to the RQ (see Section IV for a discussion), our main contribution is:

- An overview of the state-of-the-art of the current tools to support user experiments and their features.
- An overview of three behavioral science experiment tools to identify design ideas that might be adapted to programmer user studies.

In the next section (II), we revise related work. In Section V we address the threats to the validity of our work and in Section VI we present conclusions and future work.

II. Related Work

Our work builds on the work of Davis et al. [22], who presented a study in which 26 researchers were interviewed to understand their challenges when conducting programmer user studies. The study identifies several challenges and provides

This work is financed by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project UIDB/50014/2020. DOI 10.54499/UIDB/50014/2020

This research was supported by the doctoral Grant SFRH/BD/1513 66/2021 financed by the Portuguese Foundation for Science and Technology (FCT), and with funds from Portugal 2020, under MIT Portugal Program.

high-level solutions to address them. Furthermore, Davis et al. look for additional solution strategies in 10 experiment platforms (7 software engineering and 3 behavioral science experiments). However, our work focuses on the concrete tool features available on the platforms to develop user studies.

Buse et al. [1] explores both the advantages and obstacles associated with user evaluation in software engineering research. Their comprehensive analysis sheds light on the benefits of user evaluation methodologies while addressing the challenges and limitations encountered in their implementation. This work provides valuable guidance for researchers seeking to optimize user evaluation practices in software engineering studies. However, this work does not address the available experimental platforms to develop user studies and their features as we do.

Sjoeberg et al. [23] conducted a survey of controlled experiments in software engineering, thoroughly examining methodologies and outcomes in the field. However, this study only provides insights into trends, challenges, and best practices surrounding controlled experiments in SE research.

Kitchenham et al. [24] propose a set of research guidelines intended to encourage discussion and collaboration among software researchers. Their primary objective is to offer valuable assistance to researchers, reviewers, and meta-analysts as they engage in various stages of empirical study design, execution, and evaluation within the field of SE. The research guidelines are proposed to encourage debate and collaboration between researchers at the various stages of the empirical studies, but they do not characterize the SE tools as we do.

Ivie and Thain [25] discuss diverse reproducibility objectives in scientific computing. They delve into the technical challenges hindering reproducibility, review existing approaches, and highlight areas for further research. They focus on reproducibility, while our work focuses on tools to aid the execution of user studies and not necessarily on their reproducibility.

Namoun et al. [26] conducted a comprehensive review of the research issues and challenges in automated website usability evaluation tools. They provide a detailed exploration of existing tools and introduce a usability framework comprising 19 usability dimensions, examining how 10 popular web usability testing tools align with this framework. However, this work is focused on a specific domain.

Martin and Yelmo [27] offer guidance for the development of accessibility evaluation tools within the context of the Unified Software Development Process. It outlines a systematic approach to create tools that assess software accessibility. The guidance provided aids developers in ensuring that accessibility considerations are integrated seamlessly into the software development life cycle. Again, this is a very focused work.

Costa et al. [28] compared and classified 19 characteristics of existing tools designed to assist researchers across various fields in achieving reproducibility in their work. By evaluating these tools for replicating experiments, they offer guidance for researchers seeking appropriate tools to enhance the reproducibility of their experiments. However, this work is focused on reproducibility tools and not necessarily on tools to aid in user studies.

III. Experiment Tools

Davis et al. [22] conducted a literature review covering the year from 2002 to 2023 and identified seven experiment tools used in software engineering studies, including ARRESTT [11], eSEE [12], Experiment Manager [13], ExpDSL [14], [15], Ginger2 [16], K-Alpha [17] and SESE [18]. Additionally, they considered three behavioral science experiment tools, namely Gorilla [19], jsPsych [20] and LookIt [21].

Our review of prior work found no additional tools built since 2023, so we considered the same set of ten tools.

We read the published work of each tool to understand its functionalities. We have also read the available documentation/website. Based on this, we briefly describe each tool in the following paragraphs and highlight its features. We include a description whenever we describe a new feature.

A. Collected Tools

ARRESTT: The Application of Reproducible Research on Evaluation of Software Testing Technique (ARRESTT) framework offers practical support for executing and reproducing experimental studies involving software testing techniques [3], [11]. Its main features are:

- *Reuse and adaptation of user studies.* Reuse and adapt publicly available user studies to reduce the time spent developing a new study.
- *Packaging of user studies.* All the necessary information from the study is packaged (code, data, etc.) so that they can be reproduced in the future.
- *Remote experiments.* Enable remote monitoring of users' studies in order to remotely track participants' progress and reduce face-to-face interaction with participants.
- *Public sharing.* Public sharing user studies (including tasks, code, and results) to inspire others.

eSEE: The experimental Software Engineering Environment (eSEE) is a versatile framework designed to establish environments that facilitate scientific knowledge management across the entire SE experimentation process. This includes tasks such as defining, planning, executing, and packaging studies within the field of SE [12]. As the system expanded, the task of maintenance became increasingly time-consuming and impossible to continue. However, a new platform was created to maintain all the data and packages of the central integrated platform [22], [29]. Its main features are:

- *Remote experiments.*
- *Packaging of user studies.*
- *Reuse and adaptation of user studies.*

Experiment Manager: This framework supports software engineering experiments in high-performance computing (HPC) classroom environments [13]. Study participants solve programming tasks in a supervised manner using their local tools or their integrated development environment (IDE). The researcher, in this case, the professor, can monitor the experiment's progress and all the data collected. In the end, the

data collected is packaged and sent to the central web-based server for analysis. This approach provides the following main features:

- *Remote experiments.*
- *Participants registration.* Allow participants to register on the platform to facilitate interaction between researchers and potential participants. It works online, containing several user studies accessible to the platform's community.
- *Sending collected data.* Automatically send the collected data to the researcher to facilitate the data collection.
- *Packaging of user studies.*

ExpDSL: This integrated end-to-end tool provides an environment to conduct controlled human experiments using prototype domain-specific languages [14], [15]. The experiment environment is based on the Meta Programming System (MPS) and supports all steps of experimentation [30]. ExpDSL exposes the following features:

- *Remote experiments.*
- *Packaging of user studies.*

Ginger2: This platform is an integrated environment that focuses on supporting *in vitro* studies in empirical software engineering [16]. This approach, which includes hardware and software, is mainly focused on automatically collecting a variety of types of data, including three-dimensional movement, skin resistance level, mouse and keystrokes, eye traces, and videotape data. Furthermore, the experiments executed in that approach should follow a pre-determined process [22]. Given the age of this proposal, we did not find any useful characteristics to aid in current user studies.

K-Alpha: This tool for managing SE experiments is an extensible empirical software engineering computational platform aiming to address the missing functionalities in software engineering experiments using plugins [17]. The main goal of this approach is to promote the creation of reproducible experimental resources, empowering researchers to create, share, and repeat studies in the field of software testing, including techniques for test case selection and test suite minimization [17]. Its main features are:

- *Remote experiments.*
- *Participants registration.*
- *Participants preferences.*
- *Participants notification.* Notify registered participants of the opening of new user studies according to the proposed preferences of each participant.
- *Packaging of user studies.*

SESE: Simula Experiment Support Environment (SESE) [18] is a web-based platform that supports remote programmer user studies at scale and face-to-face experiments. This approach allows users to register on the platform, where they can fill in their personal data and recover the experimental data after a failure without data loss. Moreover, researchers can invite users to execute the study and benefit from a messaging system allowing the researcher to interact with the user (e.g., for instructions). Furthermore, this approach supports the managing and monitoring of all the experiment's data and the backup of the experimental data on a central server [22]. Its main features are:

- *Participants registration.*
- *Participants preferences.* Set participants' preferences to only receive notifications from the platform about the opening of user studies considered relevant.
- *Participants notification.*
- *Remote experiments.*
- *Researcher-participant communication.* Allow communication between researchers and participants for easier contact.
- *Sending collected data.*
- *Recovery of experimental data.* Automatically recover the experiment data after a failure without data loss.
- *Data backup.* Automatically backup the experimental data on a central server to prevent data loss.

Gorilla: This is an online experiment builder on which multiple researchers may design and host experiments [19]. Researchers can use its graphical user interface to configure task interfaces. Additionally, Gorilla offers a comprehensive collection of guidance materials to assist researchers in creating well-structured experiments [22]. Its main features are:

- *Reuse and adaptation of user studies.*
- *Remote experiments.*
- *Participants registration.*
- *Packaging of user studies.*
- *User studies templates.* Pre-defined templates for types of user studies to facilitate the creation of new studies.

jsPsych: This JavaScript library is an online behavioral experiment platform that structures experiments as a series of sequential steps and decisions [20]. Every experiment step offers a researcher-customizable task interface for participants to engage in specific tasks. The outcomes of these tasks are logged and can impact the subsequent stages of the experiment. jsPsych offers a core set of task interfaces tailored for behavioral science made available for the community [22]. The main features that this approach supports are:

- *Task templates.* Have pre-defined templates for task types to facilitate the creation of new tasks.
- *Participants registration.*
- *Remote experiments.*
- *Recovery of experimental data.*
- *Packaging of user studies.*

LookIt: This is a shared online platform for remote data collection on which researchers from different organizations may design and execute experiments [21]. It offers the capacity to outline the sequential actions and choices within an experiment while also equipping researchers with a range of customizable task interfaces that can be presented to participants throughout the study. Furthermore, participants are allowed to join a shared participant pool, enabling them to participate in upcoming experiments at a frequency of their preference [22]. LookIt includes the following features:

- *Remote experiments.*
- *Participants registration.*

- *Participants preferences.*
- *Participants notification.*
- *Recording user's environment.* Automatically record the user's environment (sound, screen, etc.) to allow future analysis by the researcher.
- *Data privacy level.* Choose the privacy level for the data collected.

IV. Discussion

To answer the RQ, we examined ten experiment tools: seven used by the SE community and three from behavioral science. During this evaluation, we found a total of 15 different features. In Table I, we summarize all the features and indicate which tools possess each feature. We can see that some features are available in several tools; for instance, *Remote experiments* is offered by nine tools and *Packaging of user studies* is offered by seven tools. On the other hand, *Researcher-participant communication*, *Data privacy level* and *Recovery of experimental data* are offered only by one tool each. The web-based SESE platform is the tool that covers more features, including a total of 8 of the 15 features found in all the tools.

Comparing the features of the software engineering tools with the behavioral science experiment tools, we can see that the features relating to user management and the packaging of experiences are common to both fields. On the other hand, the behavioral science experiment tools are the only ones with templates for creating tasks and user studies.

Behavioral science experiment tools offer models specifically designed to create tasks and conduct user studies. Although software engineering tools make it possible to reuse and adapt previously conducted user studies, none of these tools has built-in templates. Integrating these types of features would make it easier for the user to carry out user studies and would also serve as a guide for future tools.

V. Threats to Validity

In any research study, it is important to consider potential threats to the validity of the findings. These threats can arise from various sources and may impact the generalization and reliability of the results [31], [32]. We now discuss potential threats to the validity of our study.

The interaction of selection and treatment poses a potential threat to external validity in our analysis of experimental tools. To mitigate selection bias, we conducted a literature review covering a comprehensive time frame (from 2002 to 2023) and utilized established databases and search methodologies. Despite these efforts, the possibility remains that certain tools may not have been included in our analysis. This limitation could impact our findings regarding the landscape of experimental tools in software engineering and behavioral science.

One significant concern is the incomplete selection of features of each tool, which poses a threat to the internal validity of our analysis. During the literature review process, we selected and evaluated tools based on specific features that were documented or identifiable in the literature. However, it is possible that not all features or functionalities of each tool were captured or adequately represented in our assessment.

TABLE I
Features of experiment tools

Feature	Tools
Public sharing.	ARRESTT
Task templates.	jsPsych
User studies templates.	Gorilla
Reuse and adaptation of user studies.	ARRESTT; eSEE
Recording user's environment.	LookIt
Sending collected data.	Experiment Manager; SESE
Remote experiments.	ARRESTT; eSEE; ExpDSL; Experiment Manager; K-Alpha; SESE; Gorilla; jsPsych; LookIt;
Participants registration.	Experiment Manager; K-Alpha; SESE; Gorilla; jsPsych; LookIt
Researcher-participant communication.	SESE
Participants preferences.	K-Alpha; SESE; LookIt
Participants notification.	K-Alpha; SESE; LookIt
Packaging of user studies.	ARRESTT; eSEE; ExpDSL; Experiment Manager; K-Alpha; Gorilla; jsPsych
Data privacy level.	LookIt
Recovery of experimental data.	SESE
Data backup.	SESE; jsPsych

While analyzing the ten existing tools to aid researchers in designing and running user studies, we found 15 different features. However, the characterization of tools by features involves a certain subjectivity, which can lead to a different characterization when carried out by other researchers. Although the initial characterizations were carried out by one author, there was a discussion among the three authors to define the final list of features.

VI. Conclusion

We conducted a literature analysis of the current tools used in software engineering studies, and in three behavioral science experiment tools, we identified a list of tool features currently available to support the execution of all the stages of programmer user studies.

Our research was driven by the imperative need to facilitate and enhance the empirical evaluation of programmer-oriented tools, which is crucial for validating research prototypes and advancing production-ready solutions.

Based on this, we conclude that the features of software engineering tools and behavioral science experiment tools relating to user management and the packaging of experiences are common in both domains. On the other hand, the behavioral science experiment tools are the only ones with templates for creating tasks and user studies.

In future work, we intend to propose features addressing the barriers previously identified in [22] and the corresponding relation between features and barriers. We will also analyze the gap between current and necessary features to support user-study tools.

REFERENCES

[1] R. P. Buse, C. Sadowski, and W. Weimer, "Benefits and barriers of user evaluation in software engineering research," in *Proceedings of the 2011 ACM International Conference on Object Oriented Programming Systems Languages and Applications*, ser. OOPSLA '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 643–656. [Online]. Available: https://doi.org/10.1145/2048066.2048117

[2] L. Briand and Y. Labiche, "Empirical studies of software testing techniques: Challenges, practical strategies, and future research," *SIGSOFT Softw. Eng. Notes*, vol. 29, no. 5, p. 1–3, sep 2004. [Online]. Available: https://doi.org/10.1145/1022494.1022541

[3] F. G. De Oliveira Neto, R. Torkar, and P. D. L. Machado, "An initiative to improve reproducibility and empirical evaluation of software testing techniques," in *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, vol. 2, 2015, pp. 575–578.

[4] M. Freire, D. Costa, E. Neto, T. Medeiros, U. Kulesza, E. Aranha, and S. Soares, "Automated support for controlled experiments in software engineering: A systematic review," *Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE*, vol. 2013, pp. 504–509, 01 2013.

[5] M.-A. Storey, N. A. Ernst, C. Williams, and E. Kalliamvakou, "The who, what, how of software engineering research: a socio-technical framework," *Empirical Software Engineering*, vol. 25, no. 5, pp. 4097–4129, Sep 2020. [Online]. Available: https://doi.org/10.1007/s10664-020-09858-z

[6] A. Madugalla, T. Kanij, R. Hoda, D. Hidellaarachchi, A. Pant, S. Ferdousi, and J. Grundy, "Challenges, adaptations, and fringe benefits of conducting software engineering research with human participants during the covid-19 pandemic," *Empirical Software Engineering*, vol. 29, no. 4, p. 86, Jun 2024. [Online]. Available: https://doi.org/10.1007/s10664-024-10490-4

[7] K. Sagar and A. Saha, "A systematic review of software usability studies," *International Journal of Information Technology*, Dec 2017. [Online]. Available: https://doi.org/10.1007/s41870-017-0048-1

[8] A. A. Arechar, S. Gächter, and L. Molleman, "Conducting interactive experiments online," *Experimental Economics*, vol. 21, no. 1, pp. 99–131, Mar 2018. [Online]. Available: https://doi.org/10.1007/s10683-017-9527-2

[9] S. D. Gosling and J. A. Johnson, *Advanced methods for conducting online behavioral research.* American Psychological Association, 2010.

[10] S. Subramanian, K. De Moor, M. Fiedler, K. Koniuch, and L. Janowski, "Towards enhancing ecological validity in user studies: a systematic review of guidelines and implications for qoe research," *Quality and User Experience*, vol. 8, no. 1, p. 6, Jul 2023. [Online]. Available: https://doi.org/10.1007/s41233-023-00059-2

[11] I. Araújo, W. Silva, J. Nunes, and F. Neto, "Arrestt: A framework to create reproducible experiments to evaluate software testing techniques," 09 2016, pp. 1–10.

[12] G. H. Travassos, P. S. M. dos Santos, P. G. Mian, P. G. M. Neto, and J. Biolchini, "An environment to support large scale experimentation in software engineering," in *13th IEEE International Conference on Engineering of Complex Computer Systems (iceccs 2008)*, 2008, pp. 193–202.

[13] L. Hochstein, T. Nakamura, F. Shull, N. Zazworka, V. R. Basili, and M. V. Zelkowitz, "Chapter 5 an environment for conducting families of software engineering experiments," in *Software Development*, ser. Advances in Computers. Elsevier, 2008, vol. 74, pp. 175–200. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0065245808006050

[14] F. Häser, M. Felderer, and R. Breu, "An integrated tool environment for experimentation in domain specific language engineering," in *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE '16. New York, NY, USA: Association for Computing Machinery, 2016. [Online]. Available: https://doi.org/10.1145/2915970.2916010

[15] F. Häser, M. Felderer, and R. Breu, "Evaluation of an integrated tool environment for experimentation in dsl engineering," in *Software Quality: Methods and Tools for Better Software and Systems*, D. Winkler, S. Biffl, and J. Bergsmann, Eds. Cham: Springer International Publishing, 2018, pp. 147–168.

[16] K. Torii, K. Matsumoto, K. Nakakoji, Y. Takada, S. Takada, and K. Shima, "Ginger2: an environment for computer-aided empirical software engineering," *IEEE Transactions on Software Engineering*, vol. 25, no. 4, pp. 474–492, 1999.

[17] F. F. Silveira, R. Avancini, D. de Souza França, E. M. Guerra, and T. S. da Silva, "Towards an extensible architecture for an empirical software engineering computational platform," in *Computational Science and Its Applications – ICCSA 2021*, O. Gervasi, B. Murgante, S. Misra, C. Garau, I. Blečić, D. Taniar, B. O. Apduhan, A. M. A. C. Rocha, E. Tarantino, and C. M. Torre, Eds. Cham: Springer International Publishing, 2021, pp. 231–246.

[18] E. Arisholm, D. I. K. Sjøberg, G. J. Carelius, and Y. Lindsjørn, "A web-based support environment for software engineering experiments," *Nordic J. of Computing*, vol. 9, no. 3, p. 231–247, sep 2002.

[19] A. L. Anwyl-Irvine, J. Massonnié, A. Flitton, N. Kirkham, and J. K. Evershed, "Gorilla in our midst: An online behavioral experiment builder," *Behavior Research Methods*, vol. 52, no. 1, pp. 388–407, Feb 2020. [Online]. Available: https://doi.org/10.3758/s13428-019-01237-x

[20] J. R. de Leeuw, "jspsych: A javascript library for creating behavioral experiments in a web browser," *Behavior Research Methods*, vol. 47, no. 1, pp. 1–12, Mar 2015. [Online]. Available: https://doi.org/10.3758/s13428-014-0458-y

[21] K. Scott and L. Schulz, "Lookit (Part 1): A New Online Platform for Developmental Research," *Open Mind*, vol. 1, no. 1, pp. 4–14, 02 2017. [Online]. Available: https://doi.org/10.1162/OPMI_a_00002

[22] M. C. Davis, E. Aghayi, T. D. Latoza, X. Wang, B. A. Myers, and J. Sunshine, "What's (not) working in programmer user studies?" *ACM Trans. Softw. Eng. Methodol.*, vol. 32, no. 5, jul 2023. [Online]. Available: https://doi.org/10.1145/3587157

[23] D. Sjoeberg, J. Hannay, O. Hansen, V. Kampenes, A. Karahasanovic, N.-K. Liborg, and A. Rekdal, "A survey of controlled experiments in software engineering," *IEEE Transactions on Software Engineering*, vol. 31, no. 9, pp. 733–753, 2005.

[24] B. Kitchenham, S. Pfleeger, L. Pickard, P. Jones, D. Hoaglin, K. El Emam, and J. Rosenberg, "Preliminary guidelines for empirical research in software engineering," *IEEE Transactions on Software Engineering*, vol. 28, no. 8, pp. 721–734, 2002.

[25] P. Ivie and D. Thain, "Reproducibility in scientific computing," *ACM Comput. Surv.*, vol. 51, no. 3, jul 2018. [Online]. Available: https://doi.org/10.1145/3186266

[26] A. Namoun, A. Alrehaili, and A. Tufail, "A review of automated website usability evaluation tools: Research issues and challenges," in *International Conference on Human-Computer Interaction*. Springer, 2021, pp. 292–311.

[27] Y.-S. Martín and J. C. Yelmo, "Guidance for the development of accessibility evaluation tools following the unified software development process," *Procedia Computer Science*, vol. 27, pp. 302–311, 2014, 5th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion, DSAI 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050914000350

[28] L. Costa, S. Barbosa, and J. Cunha, "Evaluating tools for enhancing reproducibility in computational scientific experiments," in *Proceedings of the 2nd ACM Conference on Reproducibility and Replicability*, ser. ACM REP '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 46–51. [Online]. Available: https://doi.org/10.1145/3641525.3663623

[29] P. S. M. dos Santos and G. H. Travassos, "Structured synthesis method: The evidence factory tool," in *2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2017, pp. 480–481.

[30] A. Bucchiarone, A. Cicchetti, F. Ciccozzi, and A. Pierantonio, *Domain-specific Languages in Practice: With JetBrains MPS*. Springer Nature, 2021.

[31] T. Cook and D. Campbell, *Quasi-Experimentation: Design and Analysis Issues for Field Settings*. Boston: Houghton Mifflin, 1979.

[32] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer-Verlag Berlin Heidelberg, 2012, vol. 9783642290442.