The Risk-Taking Software Engineer:
A Framed Portrait

Lorenz Graf-Vlachy*†

*Institute of Software Engineering, University of Stuttgart, Stuttgart, Germany
lorenz.graf-vlachy@iste.uni-stuttgart.de
†TU Dortmund University, Dortmund, Germany
lorenz.graf-vlachy@tu-dortmund.de

Abstract—*Background:* **Risk-taking is prevalent in a host of activities performed by software engineers on a daily basis, yet there is scant research on it.** *Aims and Method:* **We study if software engineers' risk-taking is affected by framing effects and by software engineers' personality. To this end, we perform a survey experiment with 124 software engineers.** *Results:* **We find that framing substantially affects their risk-taking. None of the "Big Five" personality traits are related to risk-taking in software engineers after correcting for multiple testing.** *Conclusions:* **Software engineers and their managers must be aware of framing effects and account for them properly.**

Index Terms—**Risk-taking, framing, personality, five-factor model, Big Five**

I. Introduction

Risk-taking is prevalent in a great variety of decisions. People take risks when deciding which mating partners to choose, how to finance their homes, or which food to eat. Consequently, it is not surprising that risk-taking is one of the most extensively studied topics in a great many academic disciplines, including psychology, economics, and medicine [1].

Notably, risk-taking is also fundamental to many decisions software engineers make on a daily basis. These can be "big" decisions like choosing a software architecture or deciding which programming language to use, or "small" decisions on how well to document a minor change in code or whether to skip a test. Imagine, for instance, a software engineer who has the choice between two different libraries to accomplish a given programming task. One library has precisely the needed functionality but has not seen a new release in a while and it is unclear when and if updates and patches will become available. The other library has only limited functionality but a clear roadmap for future releases. Both options have advantages and disadvantages, but choosing the first option is likely riskier than opting for the second one.

However, there is little systematic research on what determines software engineers' risk-taking. A lot of research has focused on the management of risk, often at the project or organizational level [2], but there is hardly any work on the willingness of individual software engineers to take on risk. This is surprising given how consequential choices by software engineers can be and that a certain level of risk-taking has even been described as a desirable quality in software engineers [3].

In this paper, we attempt to remedy this shortcoming by studying two especially interesting antecedents of risk-taking.

For one, we consider an external factor, i.e., the "framing" of a decision that may influence risk-taking. For another, we study a potentially critical internal factor, i.e., the software engineer's personality. This duality of internal and external factors is particularly reasonable to consider because psychology research repeatedly demonstrated that individuals' decisions are determined both by the situation they find themselves in as well as their individual predispositions [4].

We thus attempt to answer the following research questions:

- *RQ1*: Does framing affect software engineers' risk-taking?
- *RQ2*: Does software engineers' personality affect their risk-taking?

II. Theoretical Background and Related Work

A. Framing and Risk-Taking

It has long been known that how choices are presented to individuals greatly influences the decisions they make. A particularly influential paradigm in this regard has been developed in the so-called "heuristics and biases" literature. Specifically, Tversky and Kahneman introduced the idea that the "framing" of a choice, i.e., whether it is worded in terms of potential gains or losses (while remaining logically the exact same choice), has a profound implication on respondents' level of risk-taking [5]. They studied different wordings (with the same expected outcome) and found that choices described as losses induce higher risk-taking than choices described as gains. These results have since been replicated in various studies [6]–[10].

The software engineering literature includes substantial work on heuristics and biases in general. Researchers have, for instance, found that developers are susceptible to temporal discounting [11], [12]. Other scholars found proof that developers can be substantially biased by anchoring effects [13] and that selection bias leads to project overruns [14].

Yet, there is little framing-specific research. A recent mapping study on biases in software engineering identified only three studies on framing [15]. However, they either only cursorily treat the subject or they take a much looser definition of framing, allowing for substantive differences in task descriptions (e.g., labeling desired system properties as either "requirements" or "ideas"). Further, a recent qualitative field-study on biases in software development in general mentioned

framing. However, it lumped the specific effect of framing into a larger category of biases caused by superficial thinking, neglecting the fact that framing effects tend to persist even in situations where individuals fully reason through their choices [16]. In another recent qualitative study of biases and architectural technical debt, which is closely related to risk-taking, framing was mentioned as a potential influence factor, although it was the least frequently mentioned one [17].

The most closely related work to ours is probably a study of student decision-makers who had to make requirement selection decisions. They were susceptible to a framing effect and became more risk-seeking when choosing between requirements formulated in terms of cost, compared to when choosing between requirements formulated in terms of revenue [18].

B. Personality and Risk-Taking

Although there are many different personality models in the psychology literature, the currently dominant one is arguably the five-factor model [19], [20]. As the name suggests, it comprises five personality traits, frequently also referred to as the "Big Five". These are openness to experience, conscientiousness, extraversion, agreeableness, and emotional stability (sometimes also referred to as its inverse, neuroticism).

Psychologists have repeatedly linked these personality traits to risk-taking, although with partially inconsistent findings. Some scholars, for instance, found that high extraversion and openness, combined with low neuroticism, agreeableness, and conscientiousness, is particularly predictive of risk-taking [21]. Other researchers found extraversion and agreeableness to be the key predictors of risk-taking [22].

Empirical software engineering also already has a rich tradition of studying the personality of people involved in software engineering [20], [23]. Scholars have, for example, used the Big Five personality framework to study the effect of developers' personality on the likelihood of pull-request acceptance [24]. Similarly, other studies found that committers' personality is linked to their behavior in FLOSS projects [25]. In addition, research has found that developers higher in openness to experience make more contributions to open source software projects [26]. Finally, there is extant research linking personality to programming styles [27].

At the same time, there is no research that we are aware of that explicitly attempts to link personality and risk-taking in a software engineering context.

III. EMPIRICAL SETUP

A. Stimulus Material and Measures

We took inspiration from Tversky and Kahneman's original so-called "Asian disease" problem [5], an implementation of a framing study that has been frequently used in subsequent research. To create ecological validity for our context, we adjusted the stimulus material's wording to relate it to a common software engineering problem, i.e., project delays.

Participants were randomly assigned to one of two conditions. In both conditions, participants had to make a choice between two options. The two options were substantively the same across conditions. The conditions only differed in how these options were described, or "framed". In the first condition, the options were framed as "gains", i.e., participants read about their chance of recovering time. Participants in this gain condition read the following text:

> Imagine that you are working on a software project with a deadline. You just realized that some requirements were implemented incorrectly, and you estimate that this will make you miss the deadline by 6 weeks. You think about potential remedies, and you come up with two options. You can only choose one.
>
> (A) If you reduce non-essential features, you will recover 2 weeks.
>
> (B) If you simplify the software architecture, there is a 1/3 chance that you will recover the full 6 weeks, and there is a 2/3 chance that the simplified architecture will lead to performance problems and you will not recover any time at all.
>
> Which option do you choose?

In the second condition, the options were described in terms of "losses", i.e., participants read about the delay with which they would finish the project. In this loss condition, the participants were given the following options:

> (A) If you reduce non-essential features, you will finish with a delay of 4 weeks.
>
> (B) If you simplify the software architecture, there is a 1/3 chance that you will finish the project with no delay at all, and there is a 2/3 chance that the simplified architecture will lead to performance problems and you will finish with a delay of 6 weeks.

After participants made their choice, they were forwarded to further screens on which they were asked for demographic and personality information. We captured programming experience by asking for respondents' number of years of experience [28]. We employed the widely used Ten-Item Personality Measure (TIPI) to capture respondents' personality [29]. Since this measure has been used extensively across different populations, we have no reason to doubt its suitability to assess the personality of software engineers.

B. Power Analysis and Participant Recruitment

We performed a power analysis using G*Power 3 [30] to avoid false positives and false negatives in the analysis of framing (*RQ1*) due to a potentially underpowered study. Specifically, we performed a power analysis for a z-test for proportions. We assume the relevant proportions of respondents choosing the risk-taking option to be 0.1 in the gain condition and 0.3 in the loss condition based on introspection and the stereotype that software developers overall might be

fairly risk-averse, as well as a presumed limited strength of our stimulus material. This translates into a medium effect size of $h = .52$ [31]. Conservatively specifying a two-tailed test, and setting desired alpha to 0.05 and desired power to 0.80, we obtain a critical z-value of -1.96. Further assuming an even split of participants between conditions, this implies that a sample of 124 participants is needed. Given the number of assumptions needed for a probit (or logit) power analysis, which would be needed for our analysis of personality (*RQ2*), and the limited empirical grounds we have to make them, we opted not to perform one.

To recruit participants, we obtained the contact information of all developers who made at least one commit to one of the 29 Apache open source projects that are part of the "Technical Debt Dataset" in version 2 [32]. We then identified all individuals listed as "authors" in the resulting data, and manually cleaned the data to remove duplicates and merge records for individuals who used different names (but the same email address) or different email addresses (but the same or an extremely similar name) for different commits. This required occasional judgment, and decisions about the identity of authors were made as conservatively as possible. In the end, we had a list of 1,555 unique individuals and one or more corresponding email addresses. To avoid excessive spam, we selected only one email address per person, preferring personal email addresses over professional email addresses to maximize the chance that the email address was still valid despite the person's contribution(s) to the projects being potentially already several years old. We invited all 1,555 developers to participate in our survey experiment (which was part of a larger data collection effort for multiple studies).

We assured the developers that their data would be treated confidentially and not be shared with third parties, and we pledged to donate US$ 2 per completed response to the United Nations World Food Programme [33]. We sent two reminders to reach developers that were busy at the time of the initial mailing or who had started but not completed the survey [33], including a link to an official university page confirming the authenticity of the survey because some developers responded to the initial invitation, voicing concerns about it being a scam.

In total, 165 emails bounced, allowing us to reach 1,390 developers (89.4% deliverable emails). Of this group, 194 developers started the survey, and 124 completed it. Our response rate was thus 8.9%, which is in line with that of prior studies surveying developers on GitHub. Graziotin et al., for example, reported a 7% response rate and a share of 96.6% of deliverable emails [34].

Given that our ultimate number of participants surprisingly corresponds exactly to our calculated sample size and the randomized assignment of participants to conditions lets us expect an approximately even distribution between them, we conclude that our experimental study is sufficiently powered.

IV. DATA ANALYSIS AND RESULTS

We first turn to the analysis for *RQ1*. To study whether framing had an effect on risk-taking, we compare the share of risk-taking responses between the gain and the loss condition. To this end, we employ a two-sample test for proportions (`prtest` in Stata 17.0). Out of 63 respondents in the gain condition, 7 chose the risk-taking option. Out of 61 respondents in the loss condition, 19 chose the risk-taking option. The results of the test for proportions are shown in Table I and indicate that risk-taking is statistically significantly ($p < 0.01$) higher in the loss condition. An unreported probit regression with a binary indicator of framing, as well as a two-sample Wilcoxon rank-sum test corroborate this result.

TABLE I
TWO-SAMPLE TEST FOR PROPORTIONS

Framing	Observations	Mean choice	z	p
Gain	63	.111		
Loss	61	.311		
Difference		-.200	-2.740	0.006**

Risk-averse choice coded as 0, risk-taking choice coded as 1.
$^+ p < 0.1$, $^* p < 0.05$, $^{**} p < 0.01$

To answer *RQ2*, we performed a probit regression. Aside from a binary indicator of the task framing, we included our measure of programming experience and all Big Five personality traits as independent variables. Our dependent variable was a binary indicator of whether the participant's choice was risk-taking (1) or not (0). The results are shown in Table II. The indicator for loss framing is highly significant, again confirming our earlier findings. The coefficient of programming experience is not significant. More importantly, the coefficient for conscientiousness is negative and statistically significant ($p < .05$) and the coefficient for emotional stability is positive and marginally significant ($p < .1$). However, if we (despite considerable disagreement in the applied literature as to its necessity [35]–[37]) perform a Westfall-Young correction for multiple testing (which is more efficient that the Bonferroni method [38]) to limit the family-wise error rate (using Stata's `wyoung` [39]), all coefficients for Big Five traits become insignificant (the smallest p-value being that for conscientiousness at .168).

TABLE II
PROBIT REGRESSION

Variable	Coeff.	Std. err.	z	p
Loss framing	.835	.291	2.87	.004**
Programming experience	.141	.108	1.31	.190
Openness to experience	.116	.142	0.81	.415
Conscientiousness	-.292	.130	-2.24	.025*
Extraversion	-.092	.102	-0.91	.364
Agreeableness	-.177	.128	-1.38	.168
Emotional stability	.191	.113	1.69	.092+
Constant	.119	.264	-0.96	.339

Dependent variable: Indicator of risk-aversion (0) or risk-taking (1).
$^+ p < 0.1$, $^* p < 0.05$, $^{**} p < 0.01$

V. Discussion

Software engineers overall appear to be highly risk-averse. Across conditions, only 21.0% of software engineers made a risk-taking choice despite it having the same expected outcome as the risk-averse choice. This corroborates the common stereotype of risk-averse programmers. At the same time, our results show clearly that software engineers are highly susceptible to framing effects, suggesting that the possible perception of programmers as particularly rational individuals may be misguided.

A. Implications for Research

There are several implications for research. First, we showed a framing effect for a scenario related to project delays. This raises the question for which other types of decisions or risks framing effects might exist in software engineering, and for which there might be no such effects. Similarly, our findings also raise the question if such effects are stronger or weaker for different types of roles in software development teams. In fact, one might suspect, for instance, that there could be interactive effects between task type and decision-maker role.

Second, one might wonder how to attenuate framing effects. Since the influence of framing can be considered a bias, future researchers might wish to study the effectiveness of so-called debiasing interventions in software engineers. Given that biases are mutual properties of people and tasks [40], there are two avenues for debiasing. On the one hand, one might attempt to debias individuals themselves, as has for instance been proven effective with software engineers regarding the anchoring bias [13]. On the other hand, one might study external influences as debiasing interventions. Prior research in other disciplines has, for example, found that strong warning messages may attenuate framing effects [41].

B. Implications for Practice

1) Developers: The key implication for individual developers is to realize that there might be different perspectives to take on any given situation. Explicitly constructing alternative formulations of a choice might help reach more balanced decisions that are less strongly affected by framing.

2) Managers: Managers of software projects may want to consciously consider framing in their communication with software developers. On the one hand, this is so they do not inadvertently trigger risk-taking or risk-averse behavior in developers. They may, for instance, do so by providing multiple alternative formulations of tasks or requests. On the other hand, they might use framing purposefully as a technique to increase or decrease developers' risk-taking. Further, project managers might wish to consider the idea of assigning roles to individual developers when important decisions are to be made. They might, for instance, ask one developer to think about a task in terms of gains and one in terms of losses. A discussion between the two might lead to the best outcome.

As the results for personality were not significant after correcting for multiple testing, we are hesitant to infer any implications, e.g., for team composition, from them.

VI. Threats to Validity

A. Construct Validity

As one may challenge the accuracy of our measurements, we highlight that all of our measures are established and validated scales. At the same time, we recognize that the nature of short scales like the TIPI potentially introduces substantial noise into our measurement, which might also explain why our results are not significant with regard to personality.[1]

B. Internal Validity

While we contend that our experimental study has high internal validity, our analysis on personality may suffer from deficiencies. Critically, personality is of course not randomly assigned to participants, making it possible that we missed relevant control variables that would confound our results.

In addition, the number of participants is somewhat low for a regression analysis with as many predictors as we include. Our conclusions of no personality effects might thus also be driven by low sample size and therefore be overly conservative, even though others have also reported null findings regarding developer personality [26].

C. External Validity

There is a risk that our findings may not generalize to other contexts. We studied developers involved in a limited number of large open source Java projects, with a limited response rate to our survey. Our sample is thus likely not representative of all software engineers [33]. However, we also highlight that this is possibly only a minor issue for our experimental research design, which pits one group of randomly assigned developers against another group. While these findings may thus not strictly generalize to all developers, we are nevertheless able to provide internally valid results from a sample of experienced programmers [33]. This is of course not the case for our analysis of personality, where external validity is more substantially limited.

Additionally, one might challenge whether our experiment task has external validity. For one, although this is not typically considered very problematic [43], the decision is of course hypothetical. For another, some have argued that developers make many kinds of decisions, but rarely specifically decide between two options, as they had to do in our study [44].

D. Reliability

Since we provide the stimulus material and there is no human judgment involved in data analysis, our research should be highly replicable. All data to repeat the analyses of the experiment is provided in this article. As we explicitly promised all participants that their data would not be shared with third parties, we can unfortunately not release the personality data.

---

[1]Note that the TIPI is designed to capture all facets of the Big Five with content and criterion validity with one item each, making reliability measures like Cronbach's α uninformative [42]. We thus do not report any.

VII. Future Plans

We plan to extend our work in various directions. First, we aim to collect a larger and more representative sample to replicate the study on personality effects to establish whether our null finding holds. Second, we intend to use different framing scenarios addressing different types of risky decisions software engineers may be making during their work, be it in requirements engineering, programming, testing, or other activities. Third, we strive to increase external validity by moving beyond survey experiments in favor of lab or field experiments. Specifically, we aim to study software engineering students in actual software engineering situations. Fourth, we wish to study the true interactive effects of framing and personality as well as software engineering roles to understand which kinds of software engineers are more or less susceptible to framing-induced risk-taking and which contingencies exist. Finally, we consider further extending the scope of our research by using other data collection methods such as face-to-face interviews, and by studying further influencing factors, either related to the individual developer (e.g., educational background or gender) or going beyond characteristics of the individual (e.g., organizational culture).

VIII. Conclusion

This study provides novel evidence on two types of antecedents of risk-taking in software engineers. Specifically, we show that framing has a strong influence on the risk-taking of software engineers, but we did not find reliable support for an effect of personality. We encourage future studies into the critical notion of risk-taking by software engineers.

Acknowledgment

References

[1] J. F. Yates, *Risk-taking behavior*. Chichester: Wiley, 1992.

[2] J. Masso, F. García, C. Pardo, F. J. Pino, and M. Piattini, "A common terminology for software risk management," *ACM Transactions on Software Engineering and Methodology*, vol. 31, no. 4, pp. 1–47, 2022.

[3] P. L. Li, A. J. Ko, and J. Zhu, "What makes a great software engineer?" in *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*. IEEE, 2015, pp. 700–710.

[4] R. M. Furr and D. C. Funder, "Persons, situations, and person–situation interactions," in *Handbook of personality*, O. P. John and R. W. Robins, Eds. New York: The Guilford Press, 2021, pp. 667–685.

[5] A. Tversky and D. Kahneman, "The framing of decisions and the psychology of choice," *Science*, vol. 211, pp. 453–458, 1981.

[6] N. S. Fagley and P. M. Miller, "The effect of framing on choice," *Personality and Social Psychology Bulletin*, vol. 16, no. 3, pp. 496–510, 1990.

[7] Kühberger, "The influence of framing on risky decisions: A meta-analysis," *Organizational Behavior and Human Decision Processes*, vol. 75, no. 1, pp. 23–55, 1998.

[8] A. Steiger and A. Kühberger, "A meta-analytic re-appraisal of the framing effect," *Zeitschrift für Psychologie*, vol. 226, no. 1, pp. 45–55, 2018.

[9] J. N. Druckman, "Evaluating framing effects," *Journal of Economic Psychology*, vol. 22, no. 1, pp. 91–101, 2001.

[10] M. L. DeKay, N. Rubinchik, Z. Li, and P. de Boeck, "Accelerating psychological science with metastudies: A demonstration using the risky-choice framing effect," *Perspectives on Psychological Science*, vol. 17, no. 6, pp. 1704–1736, 2022.

[11] C. Becker, R. Chitchyan, S. Betz, and C. McCord, "Trade-off decisions across time in technical debt management," in *Proceedings of the 2018 International Conference on Technical Debt*, ser. ACM Conferences, R. L. Nord, Ed. New York, NY: ACM, 2018, pp. 85–94.

[12] F. Fagerholm, C. Becker, A. Chatzigeorgiou, S. Betz, L. Duboc, B. Penzenstadler, R. Mohanani, and C. C. Venters, "Temporal discounting in software engineering: A replication study," in *2019 ACM*. Piscataway, NJ: IEEE, 2019, pp. 1–12.

[13] M. Shepperd, C. Mair, and M. Jørgensen, "An experimental evaluation of a de-biasing intervention for professional software developers," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, H. M. Haddad, R. L. Wainwright, and R. Chbeir, Eds. New York, NY, USA: ACM, 2018, pp. 1510–1517.

[14] M. Jørgensen, "The influence of selection bias on effort overruns in software development projects," *Information and Software Technology*, vol. 55, no. 9, pp. 1640–1650, 2013.

[15] R. Mohanani, I. Salman, B. Turhan, P. Rodriguez, and P. Ralph, "Cognitive biases in software engineering: A systematic mapping study," *IEEE Transactions on Software Engineering*, vol. 46, no. 12, pp. 1318–1339, 2020.

[16] S. Chattopadhyay, N. Nelson, A. Au, N. Morales, C. Sanchez, R. Pandita, and A. Sarma, "A tale from the trenches: cognitive biases and software development," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, G. Rothermel and D.-H. Bae, Eds. New York, NY, USA: ACM, 2020, pp. 654–665.

[17] K. Borowa, A. Zalewski, and S. Kijas, "The influence of cognitive biases on architectural technical debt," in *2021 IEEE 18th International Conference on Software Architecture (ICSA)*. IEEE, 2021, pp. 115–125.

[18] N. D. Fogelström, S. Barney, A. Aurum, and A. Hederstierna, "When product managers gamble with requirements: Attitudes to value and risk," in *Requirements Engineering: Foundation for Software Quality*, ser. Lecture Notes in Computer Science, M. Glinz and P. Heymans, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, vol. 5512, pp. 1–15.

[19] O. P. John, "History, measurement, and conceptual elaboration of the big-five trait taxonomy: The paradigm matures," in *Handbook of personality*, O. P. John and R. W. Robins, Eds. New York: The Guilford Press, 2021, pp. 35–82.

[20] R. Feldt, R. Torkar, L. Angelis, and M. Samuelsson, "Towards individualized software engineering," in *Proceedings of the 2008 international workshop on Cooperative and human aspects of software engineering*, L.-T. Cheng, C. de Souza, Y. Dittrich, M. John, O. Hazzan, F. Maurer, H. Sharp, J. Singer, S. E. Sim, J. Sillito, M.-A. Storey, B. Tessem, and G. Venolia, Eds. New York, NY, USA: ACM, 2008, pp. 49–52.

[21] N. Nicholson, E. Soane, M. Fenton-O'Creevy, and P. Willman, "Personality and domain–specific risk taking," *Journal of Risk Research*, vol. 8, no. 2, pp. 157–176, 2005.

[22] E. D. Joseph and D. C. Zhang, "Personality profile of risk-takers," *Journal of Individual Differences*, vol. 42, no. 4, pp. 194–203, 2021.

[23] R. Feldt, L. Angelis, R. Torkar, and M. Samuelsson, "Links between the personalities, views and attitudes of software engineers," *Information and Software Technology*, vol. 52, no. 6, pp. 611–624, 2010.

[24] R. N. Iyer, S. A. Yun, M. Nagappan, and J. Hoey, "Effects of personality traits on pull request acceptance," *IEEE Transactions on Software Engineering*, vol. 47, no. 11, pp. 2632–2643, 2021.

[25] O. H. Paruma-Pabón, F. A. González, J. Aponte, J. E. Camargo, and F. Restrepo-Calle, "Finding relationships between socio-technical aspects and personality traits by mining developer e-mails," in *Proceedings of the 9th International Workshop on Cooperative and Human Aspects of Software Engineering*. New York, NY, USA: ACM, 2016, pp. 8–14.

[26] F. Calefato, F. Lanubile, and B. Vasilescu, "A large-scale, in-depth analysis of developers' personalities in the apache ecosystem," *Information and Software Technology*, vol. 114, pp. 1–20, 2019.

[27] Z. Karimi, A. Baraani-Dastjerdi, N. Ghasem-Aghaee, and S. Wagner, "Links between the personalities, styles and performance in computer programming," *Journal of Systems and Software*, vol. 111, pp. 228–241, 2016.

[28] J. Feigenspan, C. Kastner, J. Liebig, S. Apel, and S. Hanenberg, "Measuring programming experience," in *2012 20th IEEE International Conference on Program Comprehension (ICPC)*. IEEE, 2012, pp. 73–82.

[29] S. D. Gosling, P. J. Rentfrow, and W. B. Swann, "A very brief measure of the big-five personality domains," *Journal of Research in Personality*, vol. 37, no. 6, pp. 504–528, 2003.

[30] F. Faul, E. Erdfelder, A.-G. Lang, and A. Buchner, "G*power 3: a flexible statistical power analysis program for the social, behavioral, and biomedical sciences," *Behavior research methods*, vol. 39, no. 2, pp. 175–191, 2007.

[31] J. Cohen, *Statistical power analysis for the behavioral sciences*, 2nd ed. Hillsdale, N.J.: L. Erlbaum Associates, 1988.

[32] V. Lenarduzzi, N. Saarimäki, and D. Taibi, "The technical debt dataset," in *Proceedings of the Fifteenth International Conference on Predictive Models and Data Analytics in Software Engineering*, L. Minku, F. Khomh, and J. Petrić, Eds. New York, NY, USA: ACM, 2019, pp. 2–11.

[33] S. Baltes and P. Ralph, "Sampling in software engineering research: a critical review and guidelines," *Empirical Software Engineering*, vol. 27, no. 4, 2022.

[34] D. Graziotin, F. Fagerholm, X. Wang, and P. Abrahamsson, "On the unhappiness of software developers," in *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering*, E. Mendes, Ed. New York, NY: ACM, 2017, pp. 324–333.

[35] K. J. Rothman, "No adjustments are needed for multiple comparisons," *Epidemiology*, vol. 1, no. 1, pp. 43–46, 1990.

[36] R. Bender and S. Lange, "Adjusting for multiple testing–when and how?" *Journal of clinical epidemiology*, vol. 54, no. 4, pp. 343–349, 2001.

[37] D. J. O'Keefe, "Colloquy: Should familywise alpha be adjusted? against familywise alpha adjustment," *Human Communication Research*, vol. 29, no. 3, pp. 431–447, 2003.

[38] P. H. Westfall and S. S. Young, *Resampling-based multiple testing: Examples and methods for p-value adjustment*. New York and Chichester: Wiley, 1993.

[39] D. Jones, D. Molitor, and J. Reif, "What do workplace wellness programs do? evidence from the illinois workplace wellness study," *The Quarterly Journal of Economics*, vol. 134, no. 4, pp. 1747–1791, 2019.

[40] P. Ralph, "Toward a theory of debiasing software development," in *Research in Systems Analysis and Design: Models and Methods*, ser. Lecture Notes in Business Information Processing, S. Wrycza, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, vol. 93, pp. 92–105.

[41] F.-F. Cheng and C.-S. Wu, "Debiasing the framing effect: The effect of warning and involvement," *Decision Support Systems*, vol. 49, no. 3, pp. 328–334, 2010.

[42] S. D. Gosling, "A note on alpha reliability and factor structure in the tipi." [Online]. Available: https://gosling.psy.utexas.edu/scales-weve-developed/ten-item-personality-measure-tipi/a-note-on-alpha-reliability-and-factor-structure-in-the-tipi/

[43] R. Thaler, "The psychology of choice and the assumptions of economics," in *Laboratory experimentation in economics*, A. E. Roth, Ed. Cambridge: Cambridge University Press, 1987, pp. 99–130.

[44] P. Ralph and E. Tempero, "Characteristics of decision-making during coding," in *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*, S. Beecham, B. Kitchenham, and S. G. MacDonell, Eds. New York, NY, USA: ACM, 2016, pp. 1–10.