Designing think-aloud studies to identify cognitive biases in software engineering tools: An experience report

Faith Culas, Priyanka Dhopade, Kelly Blincoe
Human Aspects of Software Engineering Lab
Waipapa Taumata Rau, University of Auckland, New Zealand
fcul804@aucklanduni.ac.nz, priyanka.dhopade@auckland.ac.nz, k.blincoe@auckland.ac.nz

***Abstract*—This experience report reflects on the process of designing and conducting think-aloud studies to explore cognitive biases in software engineering (SE) tools. While SE tools fundamentally shape how developers work, their design rarely considers diverse cognitive styles. Drawing on GenderMag's five cognitive dimensions, we conducted a series of think-aloud studies with university student developers to examine tools such as debuggers and AI-assisted coding tools and how they support or disadvantage developers with different cognitive styles. While GenderMag has been applied to some SE tools, combining it with think-aloud studies remains uncommon. This paper shares methodological insights gained during the design and execution of these studies. We discuss lessons learned about the think-aloud protocol, participants, task design, and researchers' role, highlighting challenges and strategies for mitigating bias in research design. Our recommendations aim to support researchers in designing robust studies that uncover cognitive biases in SE tools, ultimately contributing to more fairness in SE tools.**

***Index Terms*—cognitive styles, diversity and inclusion, software engineering tools, fairness, think-aloud studies, empirical studies, experiences**

I. Introduction

Software fairness is an increasingly important problem that requires a software engineering lens [1]. Like software quality, fairness concerns span the entire development lifecycle—from requirements and design to testing and verification. However, the software engineering tools that developers use to build software can themselves embed biases that shape developer experience [2], [3]. Tools are crucial to SE, and to better serve all developers equally, they must be inclusive.

Recent research has investigated "inclusivity bugs", which are barriers in SE tools that disadvantage certain developers, in tools like GitHub [2] and code reviewing tools [3]. These studies used GenderMag [4], a persona-based inspection framework that identifies inclusivity bugs through a cognitive lens. GenderMag considers five facets related to problem solving: self-efficacy, motivation, learning style, information processing, and risk attitude, represented by three personas (Abi, Pat, and Tim) to capture cognitive diversity. Using a cognitive walkthrough, researchers role-play these personas to predict potential struggles. While personas can be useful, they have limitations [5]. If experts do not represent a wide range of cognitive styles, role-playing may still overlook important inclusivity bugs. On the other hand, observing the thought processes of real users through think-aloud studies during realistic tasks provides deeper insight into how and why such barriers occur, including users' struggles, interpretations of features and strategies.

While think-aloud studies require greater time investment from both participants and researchers and may be less practical in industry settings, they offer direct access to real users' reasoning. Think-aloud protocols have proven widely effective in usability evaluation, despite debates about whether they fully capture human cognition [6]. Also, prior work has shown that theoretical assumptions about think-aloud often diverge from practice, proposing instead that we treat it as contextual, communicative speech rather than a monologue of internal thoughts [7]—a view we share as well. Our work does not re-evaluate think-aloud as a method for studying developer cognition. Instead, we report lessons learned from using think-aloud in prolonged, realistic developer tasks with student developers to surface cognitive biases embedded in SE tools. Our study shows how think-aloud reveals developers' problem-solving strategies and tool deficiencies. Through deliberate study design and careful attention to researcher and participant roles, we provide a context-specific account of how think-aloud unfolds in SE tool studies and what researchers can expect when using it for similar purposes.

In this experience report, we share insights from four iterative think-aloud studies that investigated how student developers with different cognitive styles interact with some popular SE tools—debuggers, Stack Overflow, and GitHub Copilot. In these studies, our goals were twofold: (i) generate rich qualitative evidence of cognitive barriers these tools present, and (ii) extract practical insights for improving tool design and fairness. In doing so, we found a rich set of inclusivity bugs in the studied tools, revealing biases embedded in SE tools. Given these findings, we advocate for further software fairness research using similar approaches. This paper distills lessons learned to guide future investigations into how cognitive diversity shapes interactions with SE tools.

The remainder of this paper is organized as follows: Section II gives a brief overview of the studies we have conducted so far, Section III summarises the common methodology used across the studies, Section IV presents lessons learnt and recommendations, and Section V concludes the paper.

II. Overview of Studies

Over three years (2023–2025), we conducted four iterative think-aloud studies investigating how student developers with different cognitive styles interact with three popular SE tools. Each study built on findings from the previous one, allowing us to refine our task design, facilitation approach, and analysis methods. In this paper, we synthesize lessons learned across all four studies. Below, we summarize the four studies.

Study 1 (2023): Initial study with 10 participants exploring the use of **Stack overflow**. This resulted in identifying issues related to the platform's user interface and unclear instructions. Interestingly women were more adversely affected.

Study 2 (2024/2025): The second study with 24 participants on debugger usage using the **PyCharm debugger** [8]. This resulted in 21 inclusivity bugs caused by lack of discoverability and learnability. Here too, we found trends of students with low self-efficacy, risk-averseness, low motivation, and those with process-oriented learning preferences over exploratory tinkering and selective information processing styles being disadvantaged more.

Study 3 (2025/2026): This was a follow-up study where our team made changes to some of the issues uncovered in study 2. We validated these fixes with eight participants. This resulted in insights on how the fixes help or do not help certain cognitive styles. This study is still in progress.

Study 4 (2025): Expanded to another tool, **Copilot** with 20 participants. This resulted in findings how AI-assisted coding tools surface different barriers for different cognitive styles. Currently, a manuscript reporting the results is in preparation.

III. Methodology

We provide a brief summary of the methodology below. For comprehensive methodological details, we refer readers to our full study publication [8].

Study Design and Participants. We recruited university students with minimal experience using the target tools. Participants were screened using a validated GenderMag facet questionnaire to ensure diversity across all five cognitive dimensions. Sessions lasted 60–90 minutes and comprised three phases: a 5–10 minute warm-up, a 50–60 minute observation phase, and a 10–15 minute reflective interview.

Data Collection. We collected multimodal data, including screen and audio recordings, and live observational notes. The study was conducted iteratively, beginning with pilot sessions followed by multiple rounds in the main study, with analysis between rounds. Once data saturation was reached, a final validation round confirmed that no new findings emerged. Pilot data were excluded from the results.

Data Analysis. Recordings were transcribed by linking verbalizations to on-screen actions. We conducted reflexive thematic analysis on the transcripts using both deductive and inductive approaches. To examine relationships with cognitive styles, participants were categorized using GenderMag facet scores and distributions were compared between those who experienced the most and fewest inclusivity bugs.

IV. Lessons and Recommendations

In this section, we discuss five key areas we learned through our experience in conducting think-aloud studies to identify cognitive barriers in SE tools. First, we explore how to understand and execute the think-aloud method itself. Second, we discuss the importance of intentional study design to get good quality data. Third, we address how to manage your own presence and attention during sessions as researchers. Fourth, we examine how to engage meaningfully with participants and conduct research ethically to also benefit the participants. Finally, we reflect on developing awareness of your own practice as a researcher and adapt to what works for you.

A. Understanding and Executing Think-Aloud

1) Demystifying think-aloud for novice participants: Participants were often unfamiliar with the think-aloud protocol. Simply asking participants to think-aloud was not usually enough of an explanation. For example, during an early study, a participant thought that narrating why they did something after completing the task constituted thinking-aloud. To address this, the researcher demonstrated a sample think-aloud. This simple modeling helped participants understand what we expected, and most then maintained consistent verbalization throughout their tasks.

2) The unexpected richness of silence: While verbalization was valuable, revealing data also came from non-verbal moments such as pauses, hesitations, and interactions with the tool. For example, one participant struggled with stepping through the code and used an alternative method of adding breakpoints on every single line. These silent actions signalled struggle or confusion as clearly as words. During the interview phase, we deliberately returned to these moments which helped uncover the reasoning behind their actions.

3) Managing the paradox of probing without contaminating: Timing of probes mattered. We did not always wait until the interview phase. Sometimes we intervened during observation when clarification was needed, but only strategically. For example, when a participant struggled with Python's modulus operator: *"I don't know if I'm like dumb or anything..."*, we clarified: *"Do you know how the modulus sign works? It's the remainder."* Since our focus was debugger usability, not Python competency, this intervention did not contaminate our data. We adopted the principle of probing only when it did not influence participant behaviour or contaminate the data we were collecting.

Recommendation 1

Inform and prepare participants by demonstrating what thinking-aloud means. During sessions, recognize that silence and pauses often contain meaningful cognitive work—resist the urge to fill gaps. When you do probe, ask open-ended questions carefully to clarify without leading the participants. Think-aloud is as much about listening and restraining as it is about asking questions.

B. Designing Studies Intentionally

1) The Three-Phase Structure: We adopted a three-phase format: first, a brief warmup task to familiarise participants to thinking aloud, the SE tool and the experiment workflow. We also asked questions such as 'What career do you aspire to go into after this degree?' to make the participant feel comfortable and build rapport with the researchers. Second, the observation which allowed us to observe the participant doing the study and, finally the interview, which revealed crucial context about participant strategies and frustrations. For example, a participant expressed difficulty with running test cases individually during the interview: *'The mostly challenging was is I wish if the test cases were more accessible. I have to like right click and dig for it. It took like couple of tries to figure it out. Debugging one test case is easier than the whole test suite.'* Session structure shaped data quality providing both behavioural and explanatory data.

2) Moderating difficulty - when task design shapes data quality: Task and code selection matters deeply. We learnt to choose problems in a familiar domain, so that domain knowledge wouldn't become a confounding variable. Tasks that were too easy generated silence (e.g., participants just knew what to do and found the bug), while overly difficult tasks can frustrate participants and derail their thinking or go over the time. The sweet spot was moderately challenging code, where participants needed to explore tool features to make progress, which then created the right conditions for meaningful verbalization. These tasks engaged participants throughout the full session time while naturally encouraging them to interact with and discover the tool's functionality.

3) Deliberately not knowing - removing observer bias: During recruitment, we collected data about participants' cognitive styles using the GenderMag questionnaire. While we used this information to ensure a diverse participant pool, we intentionally avoided examining individual responses until after the study was complete to observe participants without the filter of preconceptions. This practice protected our observation from confirmation bias and be open minded to all experiment data and follow-up thoroughly with the participants. Since human judgment inevitably shapes observation and interpretation, it is important for researchers to remain objective by delaying exposure to information that might bias their perceptions.

Recommendation 2

Structure your study intentionally with clear phases; warmup, observation, and reflective interviews can allow time to build rapport, capture rich data, and deepen participant insights. Calibrate task difficulty carefully to ensure meaningful interaction. Protect the integrity of your observations by strategically managing when and how you engage with participant information so that you follow-up objectively.

C. Managing Presence - Infrastructure and Attention

1) Being present - fatigue, attention, and quality: From a participant perspective, we discovered that the observation phase, the cognitively intensive part, 60 minutes is a sweet spot. Participants did not report fatigue and verbalised throughout. From a researcher perspective, three sessions in a day, was our threshold. Fatigue can affect what we notice, what we decided to probe, and how we interpreted ambiguous moments. Recordings could capture what happened, but they couldn't recover what we'd missed in real time. In the moment, when we were fresh, we could ask follow-up questions, probe deeper, clarify misunderstandings. Days later, watching recordings, we could only speculate. This taught us that being present is not just preferable; it's methodologically essential.

2) Lab Setup - simple and redundant: We used MS Teams for audio and screen recording, with IDE screen capture as backup. Redundancy saved us when recordings failed. We also took handwritten observational notes in parallel. These live notes proved critical as they captured real-time observations of confidence, confusion, and hesitation that we might overlook during later analysis. For example, one participant, after becoming comfortable with stepping through code, went silent for a while. However, our notes recorded that they were following execution well and observing variables. While some of this is visible in screen recordings, real-time notes captured nuanced observations of body language and behaviour.

Recommendation 3

It is important to manage your capacity to be fully present. We found that one researcher should limit to three sessions per day or less to maintain observation quality. Also, when you set up your lab infrastructure thoughtfully with simple, reliable systems, you can focus on participants rather than troubleshooting. This protects both your attention and observational quality.

D. Prioritizing Participant Engagement and Ethics

1) Performing under observation - acknowledging the inevitable: While most participants responded positively saying *"I like this experiment. Using debugger was useful for me"* or *"Interesting. Learning curve for sure. But getting there"*, participants were aware they were being watched, recorded, and evaluated. One participant told us directly: *"I can't read like like you're watching. I'm in a room with people."* This nervousness shaped how they interacted with the tool. This is inherent to the method itself. The research setting is artificial, observation does change behaviour, and its a limitation. Rather than trying to eliminate this effect, we learned to acknowledge it, to be open with the participants, and try to build rapport to make them feel comfortable and reminding them they could pause or stop the session at any time. This taught us that the observer effect cannot be eliminated, but it can be mitigated through openness and trust—transforming the power imbalance of being watched into collaborative inquiry.

2) Think-aloud studies benefit mutually - designing for reciprocal value: While some participants reported frustration and nervousness, many said they enjoyed the experience and learned something new. Think-aloud studies can be extractive if researchers simply take data from participants. However,

these sessions can be designed differently to reward participants through reciprocity [9]. By incorporating teaching moments, treating participant questions seriously, and acknowledging their time and effort, we created experiences where participation was genuinely valuable to students. We did not prescribe specific teaching content. Instead, at the end of sessions, we took time to show participants features they'd struggled with or misunderstood, walking through functionality together. The debrief became a space for participants to process their own experience, freely explore and play with the tool and learn something new, and for us to gather richer, more reflective commentary on what had been difficult and why. This reciprocal benefit is designable, and researchers can make participation meaningful rather than simply extractive. This transformed think-aloud from an extractive process into a mutually beneficial experience.

3) Prepare for the unexpected - designing for human participants: Designing for human participants meant accepting that it's real people and things don't go as planned always. Participants cancelled and rescheduled. Participants asked questions such as "What keyboard shortcut do I use?" We learned that treating these disruptions not as failures but as inevitable features of research with real people actually mattered.

Recommendation 4

Design research that benefits both your inquiry and your participants. Accommodate scheduling flexibly, welcome unexpected questions, and incorporate teaching moments so participation feels meaningful rather than extractive. Acknowledge that observation of participants shapes their behaviour; this a reality to recognize and work openly with the participants. When you prioritise participants, you improve both the ethical quality of your research and the quality of data you collect.

E. Developing Personal Practice as a Researcher

1) Prepared, Not Improvised: We found it helpful to prepare session workflow scripts and stick to them. Our scripts contained a flow of what to say to participants and reminders for ourselves (e.g., 'start recording now'). This put researchers at ease and ensured we did not miss details. These scripts were refined through pilot studies. We also prepared hints in advance rather than improvising on the spot. If a participant struggled to set a breakpoint, we had scripted, non-leading prompts ready (e.g., 'Think about how you can uniquely identify a line of code.'). This kept us consistent across sessions and prevented accidentally biasing the study through unscripted language.

2) Finding Efficient Practices that work for you as a Researcher: As we conducted more sessions, we developed practices that worked specifically for us. Timestamping observations in real time in our observation notes, for instance, saved time during analysis. With hours of recordings to review, researcher efficiency became a practical question of sustainability. Towards the latter part of the experiments, we had a list of inclusivity bugs/ barriers which allowed us to easily note down the barriers participants faced even while the experiments were being conducted and then verified later during transcription and analysis. Sharing these discoveries with colleagues became part of supporting qualitative research that's actually feasible to complete. Think-aloud analysis is time-intensive. Finding efficient practices that work for your workflow is helpful, whether it's timestamping observations, or developing your own note-taking system. What works for one researcher may not work for another, so along the experiments, refine your approach until you find practices that make analysis manageable and sustainable.

Recommendation 5

Preparation and efficiency go hand in hand. Prepare session scripts and prompts in advance to maintain consistency and prevent bias. Then develop efficient analysis practices that can help manage the intensive work of qualitative analysis.

V. Conclusion

Through our think-aloud studies from 2023 to 2025, we learned important lessons about how to conduct this research well. We found that the way we design tasks, how we interact with participants, and how our role as researchers all matter significantly. Think-aloud proved to be a practical and effective method for discovering cognitive inclusivity barriers in SE tools. We encourage researchers to use think-aloud studies when they want to understand how developers interact with SE tools to reveal valuable insights about tool design problems and user needs that can improve fairness in usage of SE tools.

References

[1] Y. Brun and A. Meliou, "Software fairness," in *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2018, (New York, NY, USA), p. 754–759, Association for Computing Machinery, 2018.

[2] I. Santos, J. F. Pimentel, I. Wiese, I. Steinmacher, A. Sarma, and M. A. Gerosa, "Designing for cognitive diversity: Improving the github experience for newcomers," in *2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*, pp. 1–12, IEEE, 2023.

[3] E. Murphy-Hill, A. Elizondo, A. Murillo, M. Harbach, B. Vasilescu, D. Carlson, and F. Dessloch, "Gendermag improves discoverability in the field, especially for women," in *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, ACM, 2024.

[4] M. Burnett, S. Stumpf, J. Macbeth, S. Makri, L. Beckwith, I. Kwan, A. Peters, and W. Jernigan, "Gendermag: A method for evaluating software's gender inclusiveness," *Interacting with Computers*, vol. 28, p. 760–787, Nov. 2016.

[5] N. Marsden and M. Haag, "Stereotypes and politics: Reflections on personas," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 4017–4031, ACM, 2016.

[6] M. W. Jaspers, T. Steen, C. van den Bos, and M. Geenen, "The think aloud method: a guide to user interface design.," *International journal of medical informatics*, vol. 73(11-12), p. 781–795, 2004.

[7] T. Boren and J. Ramey, "Thinking aloud: reconciling theory and practice," *IEEE Transactions on Professional Communication*, vol. 43, no. 3, pp. 261–278, 2000.

[8] F. Culas, A. Singh, A. Arankalle, P. Dhopade, and K. Blincoe, "Newcomers' experiences during debugging: A cognitive inclusivity perspective using gendermag," *Information and Software Technology*, vol. 190, p. 107932, 2026.

[9] E. Charters, "The use of think-aloud methods in qualitative research an introduction to think-aloud methods," *Brock Education Journal*, vol. 12, no. 2, 2003.