**Research Proposal: Improving Communication Between Neurodiverse Pair Programmers**

**Background:** Pair programming, a practice in Agile development and Extreme Programming involving two programmers coding together, is strongly correlated with reduced defect density, increased developer satisfaction, and persistence in computer science education [1, 2]. Benefits of pair programming can also extend to distributed development [3]. Although pair programming is a powerful tool for code quality and developer happiness and wellbeing, it has lower adoption than one might expect; one study found that only 22% of software engineers had pair programmed, citing personality conflicts as one of three key dissuading factors [4]. However, standard personality traits have not been found to correlate as strongly with pair programming productivity as expertise or task complexity [5]. What, then, drives the personality conflicts that developers cite as dissuading them from pair programming, and are those conflicts surmountable? Insights from general psychology show that *disagreement* is well-documented as a primary factor for cognitive dissonance in groups [6]. This suggests that misaligned understandings may be a bigger source of friction than personality alone in pair programming. I hypothesize that factors such as social, cultural, and neurological differences may influence such misunderstandings.

One such strong influencing factor on pair cohesion may be differing neurotypes. Communication between *neurodivergent* (e.g., ADHD, autistic) and *neurotypical* people (i.e., those who have cognitive processes considered to be the norm in their cultures) is complicated by different social, cultural, and neurological patterns. For example, the well-cited *double empathy theory* states that autistic people do not lack the ability to understand others, but that the differences between them and neurotypical people lead to pairwise misunderstandings [7]. Additionally, norms for typical workplace communication, such as turn taking, can be unclear to neurodivergent people, causing them to be unsure of when and how to talk [8]. I hypothesize that some common conflicts in pair programming, such as disagreements about the relevance of a variable to a bug [9], could be traced back to miscommunication based on neurotype incongruencies. Recent research has focused on supporting neurodivergent people, who make up an estimated 15-20% of the population [10], in the workplace. However, this research has not yet extended to the context of pair programming.

**Proposal:** Given that pair programming is an effective practice, but miscommunications based on neurotype may diminish its effectiveness or adoption, my primary research question is: *What misunderstandings arise in pair programming based on neurotype differences, and how do they affect program quality and developer happiness?* I will place an emphasis on reducing barriers for neurodivergent developers and other underrepresented groups in software engineering.

I propose an experiment studying misunderstandings arising in mixed neurotype pairs during pair programming. I will recruit a large number of professional developers from local software companies and universities, online forums, and by snowball sampling who are neurotypical, have ADHD, or are autistic. Potential participants will be prescreened to determine their software development and pair programming experiences, along with their neurotypes. Using these data, I can ensure a mix of the six types of pairs (e.g., ADHD-autistic, neurotypical-ADHD, etc.) and control for experience within pairs. Participants will attend two hour sessions, switching roles as the person at the keyboard ("driver") and the one giving high-level direction ("navigator") [5]. The computer will record participant keystrokes and program runs, and the study team will record video and take notes on apparent misunderstandings or miscommunications. Participants will complete a series of previously-validated programming tasks (such as in [5]) capturing parts of the software development lifecycle (e.g., brainstorming, debugging). In the last 15 minutes of each session, participants will be separated for a retrospective think-aloud to ascertain their comfort levels, whether they enjoyed the session, and to point to and explain a timestamp in the video that was just recorded indicating a misunderstanding they think occurred (if applicable).

**Evaluation:** To answer my question about which misunderstandings arise between developers of different neurotypes and how they affect program quality and wellbeing, I will evaluate the data via a rigorous mixed-methods approach. I will qualitatively code session videos to discover which types of misunderstandings arise, using an iterative approach and the assistance of a second researcher to assure inter-rater reliability [11]. Programs produced will be automatically evaluated for functional correctness and other quality metrics, such as complexity and readability, using established methods (such as in [12]). These metrics and

misunderstanding occurrences will be correlated to neurotype using appropriate statistical tests. My previous experience with interdisciplinary and mixed-methods analyses, developed in my two human studies that were published at a top-tier venue in software engineering, leaves me well-positioned to carry out this analysis. Critically, I will uncover more about whether neurotype mismatches result in worse code, and how we can ameliorate that by identifying which misunderstandings are most likely during certain programming tasks.

**Extensions:** This study will likely permit many follow-on research projects, making it a good long-term research direction. I am most interested in how individual program comprehension plays a role in pairwise disagreements about programs. I will identify the most common miscommunication from the main study and distill it into an individual development scenario (e.g., a concrete code review task). I will use program comprehension methods to study unpaired developers with different neurotypes during this scenario. Using the underlying factors of individual comprehension based on neurotype from this extension, as well as the results from the primary study, I will develop and evaluate a "neurotype-agnostic" framework for programming-related communication. This framework will assist knowledge transfer and individual understanding in pair contexts. I will also explore other factors, such as cultural background, which may play a role in communication conflicts reported by programmers.

This proposal supports my overall research goal of bridging communication and understanding gaps between neurotypical and neurodivergent developers. Beyond pair programming, software development is full of complex social structures which admit similar neurotype-driven misunderstandings. I plan to investigate the effects of mismatched neurotypes across social structures (e.g., a multi-person team) through observational and controlled experimental studies.

**Intellectual Merit:** To the best of my knowledge, this would be the first controlled evaluation of the effectiveness and cohesion of mixed neurotype pairs in pair programming. While previous studies have uncovered differences in communication about, or understanding of, programming tasks with respect to neurotype [8], or have evaluated the effectiveness of pairs based on attributes such as personality, attitude, or gender [5, 13, 14], none have investigated neurotype as a factor in pair composition. There is a growing body of work finding that neurotype can influence social aspects of software development [8, 15], and since pair programming is inherently social, my proposal is a desirable extension of previous work. Though this study plan is ambitious, my experience with interdisciplinary, mixed-methods, and human factors software engineering research (as demonstrated in my two peer-reviewed publications) positions me to carry it out. I will also be able to leverage my own experiences with misunderstandings in pair programming contexts, as a researcher with ADHD (see my personal statement).

**Broader Impacts:** The insights from my proposed study could substantially inform effective DEI efforts and help close the neurodivergent underrepresentation gap in software engineering. Tasks core to software engineering, such as meetings and management, have heightened barriers for neurodivergent programmers [8, 15]. The "neurotype-agnostic" framework produced by my research could be taught to developers to create more inclusive software development environments. Additionally, tools produced as a result of my research could also mitigate known barriers for neurodivergent programmers.

My research could also aid in more extensive and effective deployment of pair programming. As personality conflicts are cited as a barrier to adoption for pair programming, elucidating and repairing these personality conflicts may lead to increased use. My work could help software managers match effective programming teams during parts of the software development lifecycle.

**References**: [1] A. Cockburn and L. Williams. (2001). [2] C. Mcdowell *et al.* (2003). [3] D. Stotts *et al.* (2003). [4] A. Begel and N. Nagappan. (2008). [5] J.E. Hannay *et al.* (2009). [6] D.C. Matz and W. Wood. (2005). [7] D. E. M. Milton. (2012). [8] M. Das *et al.* (2021). [9] F. Zieris and L. Prechelt. (2020). [10] National Cancer Institute. (2022). [11] N. McDonald *et al.* (2019). [12] N. Peitek *et al.* (2021). [13] L. Thomas *et al.* (2003). [14] L. Jarratt *et al.* (2019). [15] M. Morris *et al.* (2015).