Gender Bias in Remote Pair Programming among Software Engineering Students: The twincode Exploratory Study

Amador Durán
SCORE Lab, I3US Institute
Universidad de Sevilla
Sevilla, Spain
amador@us.es

Pablo Fernández
SCORE Lab, I3US Institute
Universidad de Sevilla
Sevilla, Spain
pablofm@us.es

Beatriz Bernárdez
I3US Institute
Universidad de Sevilla
Sevilla, Spain
beat@us.es

Nathaniel Weinman
Computer Science Division
University of California, Berkeley
Berkeley, CA, USA
nweinman@berkeley.edu

Aslı Akalın
Computer Science Division
University of California, Berkeley
Berkeley, CA, USA
asliakalin@berkeley.edu

Armando Fox
Computer Science Division
University of California, Berkeley
Berkeley, CA, USA
fox@berkeley.edu

ABSTRACT

Context. Pair programming has been found to increase student interest in Computer Science, particularly so for women, and would therefore appear to be a way to help remedy the under–representation of women in the field. However, one reason for this under–representation is the unwelcoming climate created by gender stereotypes applied to engineers in general, and to software engineers in particular, assuming that men perform better than their women peers. If this same bias is present in pair programming, it could work against the goal of improving gender balance in computing. *Objective.* In a remote setting in which students cannot directly observe the gender of their peers, we aim to explore whether Software Engineering students behave differently when the *perceived* gender of their remote pair programming partners changes, searching for differences in (i) the perceived productivity compared to solo programming; (ii) the partner's perceived technical competency compared to their own; (iii) the partner's perceived skill level; (iv) the interaction behavior, such as the frequency of source code additions, deletions, validations, etc.; and (v) the type and relative frequencies of dialog messages used for collaborative behavior in a chat window. Although there are some studies on pair programming performance and gender pair combination, to the best of our knowledge there are no studies on the impact of gender stereotypes and bias *within* the pairs themselves. *Method.* We have developed an online platform (twincode) that randomly classifies students into gender–balanced groups, arranges them in pairs for remote pair programming (sharing an editor window and a chat window), and can selectively deceive one or both partners regarding the gender of the other via the use of a clearly gendered avatar. Several behaviors are automatically measured during the pair programming process, together with two questionnaires and a semantic tagging of the pairs' conversations. We will perform a series of experiments to identify the effect, if any, of possible gender bias in remote pair programming interactions. Students in the control group will have no information about their partner's gender; students in the treatment group will receive such information but will be selectively deceived about their partner's true gender. To analyze the data, apart from checking reliability of questionnaire data using Cronbach's alpha and Kaiser criterion, for each response variable we will (i) compare control and experimental groups for the score distance between two in–pair tasks; then, using the data from the experimental group only, we will (ii) compare scores using the partner's perceived gender as a within–subjects variable; and (iii) analyze the interaction between the partner's perceived gender (within–subjects) and the subject's gender (between–subjects). For the (i) and (ii) analyses we will use t–tests, whereas for the (iii) analyses we will use mixed–model ANOVAs.

CCS CONCEPTS

• **General and reference** → **Empirical studies**; • **Software and its engineering** → **Agile software development**.

KEYWORDS

Gender Bias, Pair Programming, Remote Pair Programming, Distributed Pair Programming, Software Engineering Education

1 INTRODUCTION

Pair programming is an increasingly popular collaboration paradigm that has been shown to be an effective tool in Computer Science education as measured by positive influence on grades, class performance, confidence, productivity, and motivation to stay

[6], especially for women [19, 24]. In pair programming, two part-
ners work closely together to solve a programming task. As such,
their ability to engage with each other is key. However, these in-
teractions are influenced by implicit gender bias [12, 18], such as
assuming women are less technically competent [18]. This is a
widely observed phenomenon even in highly–structured settings
[6, 13]. Social sciences research indicates that one's behavior of
an individual is affected by the behavior of their peers [8]. There-
fore, implicit gender bias based on perception of peers may have
effects on one's behavior, potentially influencing pair programming
experience.

In this work, in a non–colocated (i.e. remote) environment in
which the gender of the peers cannot be directly observed, our
goal is to explore whether Software Engineering students change
their behavior when the *perceived* gender of their remote pair pro-
gramming partners changes from man to woman or vice versa.
Note that, while we recognize that many students may identify as
neither men nor women, our initial exploration focuses primarily
on interactions between students who identify as one of these, so
that we can better align our findings with the existing literature on
implicit gender bias. The potential biases in interactions involving
gender-fluid, non-gender-conforming, or nonbinary students is a
rich and complex topic deserving its own subsequent study.

To achieve our goal, we plan to search for differences not only
in the perceived productivity of pair programming compared to
solo programming, the partner's perceived technical competency
compared to their own, and the partners' perceived skill level, but
also in the interaction behavior, i.e. the frequency of source code
additions, deletions, validations, etc., and the type and relative
frequency of dialog messages used for collaborative behavior.

To get early feedback on the infrastructure supporting our pro-
posal, we ran two pilot studies, one at each university, with a limited
number of students, where we could check the comprehensibility of
the questionnaires used to gather subjective data, the applicability

of the message tagging (described in Section 2), and the capabilities
of the twincode platform, which is briefly described below.

1.1 The twincode platform

To support our study, we have developed the twincode remote pair
programming platform, which manages the registration of students,
the random allocation to gender-balanced groups, the random allo-
cation into pairs, the random assignment of programming exercises
to pairs, and the automatic collection of interaction metrics and
dialog messages.

As shown in Figure 1, twincode offers a source code editor
where the students concurrently develop the solution to a proposed
exercise and can validate it against several test cases. It also offers
a chat window, where they can collaborate to solve the exercise.
Note that a gendered avatar is displayed for the student in the
experimental group only (right), but not for the one in the control
group (left).

1.2 Related Work

Several literature reviews [10, 15, 21] have compiled the empiri-
cal research on using pair programming in higher education, with
[6] being focused on distributed pair programming from a teach-
ing perspective. By means of controlled experiments, remote and
co–located pair programming are compared in [1, 22], showing
comparable results. In most of the cases, the analyzed variables
are related to performance in terms of time, quality, or code tests
passed. Students perceptions have been also analyzed in terms of
confidence, satisfaction, motivation, or personality [20].

Table 1 summarizes the empirical studies on the influence of
gender in pair programming, including findings such as (i) same–
gender pairs are more "democratic"; (ii) women working in pairs
were more confident than those working solo; and (iii) in mixed-
gender, pairing women particularly do not benefit [15]. Although

Figure 1: twincode user interface for control group (left) and experimental group (right)

Reference	Object of study	Metrics	Findings
Katira et al. [14]	Compatibility of student pair programmers	Web–based peer evaluation survey that required the students to evaluate the contributions of their partner and the pair compatibility as perceived by the student	Students are compatible with partners whom they perceive of similar skill. Mixed-gender pairs are less likely to report compatibility.
Choi et al. [4]	PP gender combinations	Productivity, quality of source code, compatibility and communication between pairs	Significant differences in the levels of pair compatibility and communication between the same gender pair type, woman–woman and man–man.
Gómez et al. [9]	PP gender combinations	Productivity	Similar productivity rates for the three gender pair combinations. The programming assignments had a significant impact on the productivity. Greater variability of productivity rates with mixed gender pairs (man–woman) was observed.
Jarrat et al. [13]	PP gender combinations	Weekly attendance, work accomplished during lab and perceptions of productivity	Students who were randomly assigned a woman as a partner (rather than a man as a partner) attended class more often, were more confident that the solution was correct, and more confident in the finished product that they created. However, being assigned a woman as a partner was also associated with completing a smaller percentage of the assignment.

Table 1: Empirical studies about gender in pair programming

such studies reveal that gender seems to be a key factor in pair programming, none of them study gender bias in pair programming.

Many factors other than gender may affect the outcomes of remote programming sessions [2, 23]. Previous research on productive pairing looked at factors such as skill levels, autonomy in choosing one's partner [25], and different personalities [11]. Nevertheless, the work on gender composition of pairs found conflicting results about whether same-gender or mixed-gender pairings are more effective [3, 4, 12, 16]. One possible explanation is that gender correlates with other dimensions that may affect the pairs' collaboration, but these correlations may vary between different environments. For example, women in a class may, on average, have higher skill level than men because they had to face more societal barriers to enter the class. On the other hand, they may, on average, have lower skill level if women with no background are more actively recruited.

2 RESEARCH QUESTIONS

Our study is based on the hypothesis that gender bias will lead to observable differences based on subjects' perceptions of the gender of their partners, i.e., they will score men and women differently for similar tasks and also behave differently depending on the perceived gender of their partner. To study our hypothesis, we plan to apply methodological triangulation [7], using several methods to collect data and approaching a complex phenomenon like human behavior from more than one standpoint [5]. In our case, three different data sources will be used: questionnaires completed by the subjects, data collected automatically by the twincode platform, and data produced by two different experimenters analyzing and tagging dialog messages and cheking interrater agreement using Cohen's kappa coefficient [17].

With respect to the data collected using questionnaires, our research questions are:

RQ$_1$ In remote pair programming, does gender bias affect perceived productivity compared to solo programming?

RQ$_2$ In remote pair programming, does gender bias affect the partner's perceived technical competency compared to one's own technical competency?

RQ$_3$ In remote pair programming, does gender bias affect how partners' skills are perceived?

With respect to the data automatically collected by the twincode platform—which could be increased in the future—our research question is:

RQ$_4$ In remote pair programming, does gender bias affect the frequencies or relative frequencies with which each partner produces source code additions, source code deletions, successful validations, failed validations, and dialog messages?

The manual semantic tagging of the dialog messages classifies each message into two orthogonal dimensions. The first dimension uses the 13 tags proposed in [19] (tags from S to O in Table 2). The second dimension classifies each message as formal or informal. With respect to this data source, our research questions are:

RQ$_5$ In remote pair programming, does gender bias affect the frequency or relative frequency of the different types of dialog messages?

RQ$_6$ In remote pair programming, does gender bias affect the relative frequency of formal and informal dialog messages?

3 VARIABLES

In this section, we describe all the variables we will consider in our study. Note that depending on the development of the twincode platform, more automatically measured dependent variables could be added in the future.

When used, abbreviations are enclosed in parentheses after variables' names.

Tag	Description	Examples
I	Informal	*LOL! Hahaha!*
F	Formal	All messages except informal
S	Statement of information or explanation	*We need to create a program for kids to learn math*
U	Opinion or indication of uncertainty	*Unsure how to add strings together*
D	Explicit instruction	*Wait put the if back*
SU	Polite or indirect instruction	*Maybe we can do if user choice = +*
ACK	Acknowledgement	*Oh ok gotcha*
M	Meta–comment or reflection	*Hmmm*
QYN	Yes/no question	*Can the answer be negative?*
QWH	Wh– question (who, what, where, when, why, and how)	*How do I take in their input?*
AYN	Answer to yes/no question	*Yea*
AWH	Answer to wh– question	*The program should be able to generate erroneous questions*
FP	Positive task feedback	*Oh nice*
FNON	Non–positive task feedback	*Thats weird*
O	Off–task	*Wow its sweet in this room*

Table 2: Dialogue tags from [19] augmented with orthogonal informal/formal tags

3.1 Independent Variables

group nominal factor representing the group (experimental or control) subjects are randomly allocated to.

time nominal factor representing the moment (t_1 and t_2) in which the first and second in–pair tasks are performed by the subjects.

induced partner's gender (ipgender) nominal factor representing the induced partner's binary gender (man or woman for the experimental group, and none for the control group, in which gender is not revealed) during the in–pair tasks. This variable, which is directly related to the gender bias treatment, is operationalized by means of the instructions provided at the beginning of the tasks ("…work with your partner. She is …") and by the gendered avatar that is visible during the in–pair tasks for the experimental group and that is swapped between tasks. Subjects in the control group receive no treatment, i.e., they do not see any information about the gender of their partners in any way, neither textual nor graphical.

gender nominal factor representing subject's gender, which may be man, woman, or any other option as freely expressed in the demographic form during registration.

3.2 Dependent Variables

The response variables measured using questionnaires containing 0–10 linear numerical response items are the following:

Perceived productivity compared to solo programming (pp) interval variable measuring the subject's perceived productivity compared to solo programming after each in–pair task (see RQ_1). Low values correspond to better solo programming productivity, i.e., "solo programming would have been more productive than pair programming", whereas high values correspond to better pair programming productivity, i.e. "pair programming has been more productive than solo programming".

Perceived partner's technical competency compared to their own (pptc) interval variable measuring the subject's partner's perceived technical competency compared to their own after each in–pair task (see RQ_2). Low values correspond to higher subject's productivity, i.e., "I have been more productive than my partner", whereas higher values correspond to high partner's productivity, i.e. "My partner has been more productive than me".

Compared partners' skills (cps) interval variable measuring whether the subject perceived better skills in their first or second partner in the in–pair tasks (see RQ_3). Low values correspond to the first partner, i.e., "My first partner was a better partner than my second partner", whereas high values correspond to the second partner, i.e. "My second partner was a better partner than my first partner". In the case of the experimental group only, this variable is transformed after collection using an R script in such a way that low values correspond to the partner perceived as a man, and high values to the partner perceived as a woman, in order to analyze whether there is a gender bias in the scoring.

Apart from the variables described above, the questionnaires will also include questions about the perceived gender of their parners at each task. The corresponding variable is described below:

perceived partner's gender (ppgender) nominal factor representing the subject's perception of their partner's gender (woman, man, I don't know, or I don't remember) at each in–pair task.

The response variables automatically collected by the `twincode` platform and related to the interaction during the in–pair programming exercises (see RQ_4) are listed below. Every variable v represents a frequency, i.e., a count, and its associated relative frequency is computed with respect to the the sum of the frequencies of the two subjects in a pair. For example, let us suppose that subjects i and j are the two members of a pair, and v_i and v_j are the corresponding values of the v variable. In this case, the relative frequencies for each subject would be $\frac{v_i}{v_i + v_j}$ and $\frac{v_j}{v_i + v_j}$, respectively.

source code additions (sca) Ratio scale variable representing the count of characters added by a subject to the source code window during an in–pair task.

source code deletions (scd) Ratio scale variable representing the count of characters deleted by a subject from the source code window during an in–pair task.

successful validations (okv) Ratio scale variable representing the count of successful validations of the source code performed by a subject during an in–pair task.

unsuccessful validations (kov) Ratio scale variable representing the count of unsuccessful validations of the source code performed by a subject during an in–pair task.

dialog messages (dm) Ratio scale variable representing the count of dialog messages sent by a subject during an in–pair task.

The response variables related to the manual tagging of the dialog messages (see RQ_5 and RQ_6) correspond to the tags in Table 2 and are listed below. Every variable represents a frequency, i.e., a count, and its associated relative frequency is computed with respect to the number of dialog messages generated by the subject during an in–pair task, which is defined by the dm variable specified above.

i Ratio scale variable representing the count of informal messages generated by a subject during an in–pair task.

f Ratio scale variable representing the count of non–informal, i.e. formal, messages generated by a subject during an in–pair task.

s Ratio scale variable representing the count of statement of information or explanation messages generated by a subject during an in–pair task.

u Ratio scale variable representing the count of opinion or indication of uncertainty messages generated by a subject during an in–pair task.

d Ratio scale variable representing the count of explicit instruction messages generated by a subject during an in–pair task.

su Ratio scale variable representing the count of polite or indirect instruction messages generated by a subject during an in–pair task.

ack Ratio scale variable representing the count of acknowledgment messages generated by a subject during an in–pair task.

m Ratio scale variable representing the count of meta–comment or reflection messages generated by a subject during an in–pair task.

qyn Ratio scale variable representing the count of yes/no question messages generated by a subject during an in–pair task.

qwh Ratio scale variable representing the count of wh- question (who, what, where, when, why, and how) messages generated by a subject during an in–pair task.

ayn Ratio scale variable representing the count of answer to yes/no question messages generated by a subject during an in–pair task.

awh Ratio scale variable representing the count of answer to wh- question messages generated by a subject during an in–pair task.

fp Ratio scale variable representing the count of positive task feedback messages generated by a subject during an in–pair task.

fnon Ratio scale variable representing the count of non–positive task feedback messages generated by a subject during an in–pair task.

o Ratio scale variable representing the count of off–task messages generated by a subject during an in–pair task.

3.3 Confounding Variables

The confounding variables that will be controlled during the experiment are the following.

Subject's technical skills To control the variability caused by each subject on their partner, pairs are kept the same during the entire experiment, although the subjects are not informed about this fact until the end of the experiment. Ideally, this would make the conditions of the two in–pair tasks the same except for the programming exercises (see below) and for the perceived gender in the case of the experimental group.

Programming exercises In order to avoid potential differences among the programming exercises used during in–pair tasks, they are of similar difficulty and are randomly assigned.

4 PARTICIPANTS

As we have done in the pilot studies, our plan is offering our students the possibility to participate in the twincode study. At the University of Seville, the participants will be 3rd–year students of the Degree in Software Engineering. We expect the number of participants to be around 150–200.

At the University of California Berkeley, the participants will be students enrolled in the 1st and 2nd semester Computer Science courses. In this case, we expect the number of participants to be around 100–300.

5 EXECUTION PLAN

We plan to perform a baseline experiment at the University of Seville in the first semester of the 2021–2022 academic course, and then, replicate the experiment at the University of California Berkeley at the beginning of the second semester. At each location, the experimental material provided to the students will be localized, i.e., in Spanish in Spain and in English at the USA, and the translation will be carefully checked by bilingual experimenters.[1]

Independently of being the baseline or the replication, the execution plan of the twincode study consists of the steps described below.

5.1 Recruitment

In this initial step, we plan to motivate the students to voluntarily participate in the study as an interesting experience in remote pair programming but without mentioning that the main goal is to study the effect of gender bias. We also remark that for the purpose of the study, they must remain anonymous to their partners, so they must neither mention nor ask any personal information.

The interested students must register in the twincode platform providing some demographic data and accepting the participation conditions.

5.2 Training

One week or so before the experiment execution, we plan to provide a short training session about the twincode platform, so students become familiar with it and any concerns about how to use the platform were dispelled, thus reducing any potential anxiety for using a new platform.

---

[1]At the time of writing, discussions are underway for another replication at the Northern Technical University at Ecuador. If this replication is eventually carried out, the experimental material will also be adapted to the local variant of Spanish if deemed necessary.

Figure 2: Experimental process (subject allocation to groups)

Figure 3: Experimental process (tasks)

5.3 Experiment Execution

For experiment execution, all registered students must log into the twincode platform, which will automatically allocate them into the control and experimental groups. This allocation is random and gender–balanced, i.e., the proportion of men and women in each group is as close as possible. Once all students are allocated to groups, they are randomly allocated into control–experimental pairs, as shown in Figure 2.

After subject allocation, the pairs are presented a programming exercise that they must solve collaboratively (labeled as Task #1 in Figure 3) in the twincode platform. They are given 20 minutes to do it, and in case they finished earlier, another exercise of increasing difficulty is presented. At the end of the 20–minute period, they are asked to individually fill in a questionnaire about the perceived productivity compared to solo programming and about the perceived partner's technical competency compared to their own. They are given 10 minutes to fill in the questionnaire.

After filling the first questionnaire, the students are presented another programming exercise to be solved individually in 10 minutes (Task #2 in Figure 3). As in the previous task, another exercise of increasing difficulty is presented if they finish earlier. The main purpose of this individual task is to make students forget about their first partners, i.e. their way of writing dialog messages or source code, so they do not recognize them in the second in–pair task.

After the individual task, pairs are presented again a new collaborative programming exercise that they must solve in similar conditions to the exercise in Task #1. In this second in–pair exercise, the avatar gender (which in this initial study will always indicate binary gender) is swapped with respect to the first exercise for the subjects in the experimental group. Note that pairs are kept the same in order to reduce the variability due to the subjects themselves, which could possibly have had a confounding effect in case of a new pair allocation for Task #3.

Once Task #3 is finished, students are asked to fill in the same questionnaire than they filled after Task #1, and another questionnaire comparing the perceived genders and skills of the first and second partners. They are given 15 minutes for responding both questionnaires.

Finally, they are informed about the actual goal of the study and that the pairs remained the same during the experiment. At this moment, they are allowed to withdraw their data if they wish to do so.

5.4 Data Analysis

During the manual tagging of the dialog messages, all pairs in which the gender of any of the peers is disclosed in any way are excluded from the data analysis. Then, before analyzing response variables, the internal consistency of the questionnaire data is checked using Cronbach's alpha and Kaiser criterion. After that, for every dependent variable v, we compute the distance between the two in–pair tasks as the absolute value of the difference, i.e. $|v(t_2) - v(t_1)|$. Ideally, this distance should be lower for the students in the control group (no information about partners' genders) than for those in the experimental group (with two different perceived partners' genders at t_1 and t_2). Therefore, for every variable, we perform a t–test to detect distance differences between the groups.

Then, using the data from the experimental group only, we perform a t–test to detect differences in the scores of every dependent variable between perceived partner's gender, i.e. to detect differences in the scores when partners are perceived as men vs. as women. Finally, to detect a potential interaction between the perceived partner's gender and the subject's gender, we perform a mixed–model ANOVA with the perceived gender as a within–subjects variable and subject's gender as a between–subjects variable. As complementary analyses, we also study (i) the correlation between the induced and the perceived gender for the subjects in the experimental group, and the distribution of the perceived gender (if any) in the control group; and (ii) the potential cultural impact of the different locations at which the experiment is carried out.

All the data analysis will be performed using R scripts, that will be available in a public repository together with the datasets in the corresponding laboratory package.

REFERENCES

[1] Ahmad Al-Jarrah and Enrico Pontelli. 2016. On the effectiveness of a collaborative virtual pair-programming environment. In *International Conference on Learning and Collaboration Technologies*. Springer, 583–595.

[2] E. A. Chaparro, Aybala Yuksel, Pablo Romero, and Sallyann Bryant. 2005. Factors Affecting the Perceived Effectiveness of Pair Programming in Higher Education. In *Proceedings of the 17th Workshop of the Psychology of Programming Interest Group*.

[3] Kyungsub S. Choi. 2013. Evaluating Gender Significance within a Pair Programming Context. In *Proceedings of the 2013 46th Hawaii International Conference on System Sciences (HICSS '13)*. IEEE Computer Society, USA, 4817–4825. https://doi.org/10.1109/HICSS.2013.209

[4] Kyungsub Stephen Choi. 2015. A comparative analysis of different gender pair combinations in pair programming. *Behaviour & Information Technology* 34, 8 (2015), 825–837. https://doi.org/10.1080/0144929X.2014.937460 arXiv:https://doi.org/10.1080/0144929X.2014.937460

[5] Louis Cohen, Lawrence Manion, and Keith Morrison. 2018. *Research Methods in Education* (8th ed.). Routledge.

[6] Bernardo José da Silva Estácio and Rafael Prikladnicki. 2015. Distributed Pair Programming: A Systematic Literature Review. *Information and Software Technology* 63 (2015), 1–10.

[7] N.K. Denzin. 2006. *Sociological Methods: A Sourcebook* (5th ed.). Aldine Transaction.

[8] Dean Eckles, René F. Kizilcec, and Eytan Bakshy. 2016. Estimating peer effects in networks with peer encouragement designs. *Proceedings of the National Academy of Sciences* 113, 27 (2016), 7316–7322. https://doi.org/10.1073/pnas.1511201113 arXiv:https://www.pnas.org/content/113/27/7316.full.pdf

[9] O. Gómez, M. Solari, C. Calvache, and A. Ledezma-Carrizalez. 2017. A Controlled Experiment on Productivity of Pair Programming Gender Combinations: Preliminary Results. In *Proceedings of the XX Ibero–American Conference on Software Engineering*. 197–210.

[10] Brian Hanks, Sue Fitzgerald, Renée McCauley, Laurie Murphy, and Carol Zander. 2011. Pair programming in education: A literature review. *Computer Science Education* 21, 2 (2011), 135–173.

[11] Jo E. Hannay, Erik Arisholm, Harald Engvik, and Dag I.K. Sjoberg. 2010. Effects of Personality on Pair Programming. *IEEE Transactions on Software Engineering* 36, 1 (2010), 61–80. https://doi.org/10.1109/TSE.2009.41

[12] Sarah I. Hofer. 2015. Studying Gender Bias in Physics Grading: The role of teaching experience and country. *International Journal of Science Education* 37, 17 (2015), 2879–2905. https://doi.org/10.1080/09500693.2015.1114190 arXiv:https://doi.org/10.1080/09500693.2015.1114190

[13] Lindsay Jarratt, Nicholas A. Bowman, K.C. Culver, and Alberto Maria Segre. 2019. A Large-Scale Experimental Study of Gender and Pair Composition in Pair Programming. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education* (Aberdeen, Scotland Uk) *(ITiCSE '19)*. Association for Computing Machinery, New York, NY, USA, 176–181. https://doi.org/10.1145/3304221.3319782

[14] N. Katira, L. Williams, and J. Osborne. 2005. Towards increasing the compatibility of student pair programmers. In *Proceedings. 27th International Conference on Software Engineering, 2005. ICSE 2005*. 625–626. https://doi.org/10.1109/ICSE.2005.1553618

[15] K. Kaur Chahal, A. Kaur, and M. Saini. 2021. *Research and Evidence in Software Engineering: From Empirical Studies to Open Source Artifacts*. Taylor & Francis Group, Chapter Empirical Studies on Using Pair Programming as a Pedagogical Tool in Higher Education Courses: A Systematic Literature Review, 251–287.

[16] Sandeep Kaur Kuttal, Kevin Gerstner, and Alexandra Bejarano. 2019. Remote Pair Programming in Online CS Education: Investigating through a Gender Lens. In *2019 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 75–85. https://doi.org/10.1109/VLHCC.2019.8818790

[17] J.R. Landis and G.G. Koch. 1977. The Measurement of Observer Agreement for Categorical Data. *Biometrics* 33, 1 (1977), 159–174.

[18] Martell, D. M. Lane, and C. Emrich. 1996. Male-female differences: A computer simulation. *American Psychologist* 2, 51 (7 1996), 157–158. https://doi.org/10.1037/0003-066X.51.2.157

[19] Fernando J. Rodríguez, Kimberly Michelle Price, and Kristy Elizabeth Boyer. 2017. Exploring the Pair Programming Process: Characteristics of Effective Collaboration. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (Seattle, Washington, USA) *(SIGCSE '17)*. Association for Computing Machinery, New York, NY, USA, 507–512. https://doi.org/10.1145/3017680.3017748

[20] Norsaremah Salleh, Emilia Mendes, and John Grundy. 2014. Investigating the effects of personality traits on pair programming in a higher education setting through a family of experiments. *Empirical Software Engineering* 19, 3 (2014), 714–752.

[21] Norsaremah Salleh, Emilia Mendes, John Grundy, and Giles St J Burch. 2010. The effects of neuroticism on pair programming: an empirical study in the higher education context. In *Proceedings of the 2010 ACM-IEEE international symposium on empirical software engineering and measurement*. 1–10.

[22] David Stotts, Laurie Williams, Nachiappan Nagappan, Prashant Baheti, Dennis Jen, and Anne Jackson. 2003. Virtual teaming: Experiments and experiences with distributed pair programming. In *Conference on Extreme Programming and Agile Methods*. Springer, 129–141.

[23] Lynda Thomas, Mark Ratcliffe, and Ann Robertson. 2003. Code Warriors and Code-a-Phobes: A Study in Attitude and Pair Programming. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education* (Reno, Navada, USA) *(SIGCSE '03)*. Association for Computing Machinery, New York, NY, USA, 363–367. https://doi.org/10.1145/611892.612007

[24] Linda L. Werner, Brian Hanks, and Charlie McDowell. 2004. Pair-Programming Helps Female Computer Science Students. *J. Educ. Resour. Comput.* 4, 1 (March 2004), 4–es. https://doi.org/10.1145/1060071.1060075

[25] S. Xinogalos, M. Satratzemi, A. Chatzigeorgiou, and D. Tsompanoudi. 2017. Student perceptions on the benefits and shortcomings of distributed pair programming assignments. *2017 IEEE Global Engineering Education Conference (EDUCON)* (2017), 1513–1521.