Acceptance and Assessment in Student Pair-Programming: A Case Study

https://doi.org/10.3991/ijet.v16i09.18693

Ramón Ventura Roque-Hernández
Autonomous University of Tamaulipas, Nuevo Laredo, México

Sergio Armando Guerra-Moya
Autonomous University of Nuevo León, Monterrey, México

Frida Carmina Caballero-Rico (✉)
Autonomous University of Tamaulipas, Ciudad Victoria, México
fcaballer@uat.edu.mx

Abstract—The learning of programming is a complex process that may lead to high failure and dropout rates. Pair Programming is a software development technique that involves two individuals working side-by-side using the same computer. It has proven to be helpful in the process of programming learning. This paper analyses pair programming's acceptance and assessment in novice programming university courses considering participants' gender, previous programming experience and programming enjoyment. The participants were 80 students from three different cohorts enrolled in the IT undergraduate program at a Mexican State University. We designed and administered a questionnaire to collect data after the pair programming exercises were performed. For data analysis, we used SPSS 24, Mann-Whitney, Kruskal-Wallis and Jonckheere-Terpstra statistical techniques. Descriptive and comparative results showed a significant increasing monotonic trend in the acceptance of Pair Programming as students' preference towards programming increased (standardized statistic = 3.20, p = 0.00, Kendall's b = 0.30, p = 0.001). There were no other statistically significant results. We found Pair Programming helpful and easy-to-implement in university courses. Students, on the other hand, positively accepted and assessed this learning approach. We recommend Pair Programming adoption in university courses as part of a broader strategy to engage students in the effective learning of programming.

Keywords—Higher education, students, pair programming, universities

1 Introduction

The learning of programming is not an easy process [1], especially for novice programmers [2]. Research shows that students have trouble understanding and passing programming courses to such an extent that they may drop out of school [3]. On the

other hand, the teaching of programming is also complex and has been identified as one of the seven major challenges in computer science [4]. Although many pedagogical approaches have been used in the teaching of Programming in the last 30 years [5] and new teaching techniques have been continuously proposed and researched, we still do not have a deep understanding of them. Since coding activities are present from kindergarten through university [6], programming is a subject that poses serious challenges for students and teachers. This is perhaps because of the variables and conditions of effective application of these didactic strategies in the teaching of programming have not been exhaustively studied and implemented in practice.

Pair Programming is a software development technique introduced during the mid-90s as a component of Extreme Programming. It is a useful methodology to create software [7]. This practice involves two individuals working side-by-side using the same computer and collaborating through analyzing, designing, coding, and testing in a shorter period. Pair Programming has improved functionality and productivity in the software industry [8]. Some factors contributing to its successful implementation includes the participants' expertise, the project's preparation, and the perceived quality of the solution.

In the educational context, Pair Programming has become a popular pedagogical tool in beginner programming courses due to its benefits [9], especially when implemented in early semesters face-to-face [10], virtual modalities, as a single approach or as part of a blended learning model [11]. Smith, Giugliano and Deorio [12] report that students who work in pairs in beginner courses could get higher grades in more advanced courses. Their results show that students with the lowest grades were the most benefited. In higher education, Pair Programming studies have also considered aspects such as the students' semester and the difficulty level in the course.

Pair Programming as a pedagogical tool is an easy implementation technique that has proven to be effective in developing functional software. It can improve communication, knowledge sharing and human relations. Its study in the university context is relevant because it can help students to learn more in a better environment, reducing the high rates of failure and dropout that observed in technology careers. Pair Programming supports students in their transition through programming courses by emphasizing human aspects and communication among participants, by improving student performance, retention, and motivation. However, a deep understanding of the students' experience in Pair Programming remains a challenge for researchers [13]. If Pair Programming were well accepted by students, it could be implemented as a regular strategy to avoid failing and dropping out. Moreover, students could continue and conclude their professional preparation which would be highly beneficial for our society.

In this respect, we find it is relevant to study students' acceptance and the benefits they perceive about Pair Programming because it is them, who learn by developing software in pairs. Zikmund, Babin, Carr and Griffin [14] say that perceptions are people's dispositions to respond to their surrounding world. Sun, Marakas and Aguirre-Urreta [15] say that perceptions are a way of considering, understanding or interpreting a phenomenon because they are one of the multiple factors that influence thinking skills. That is why perceptions about programming methodologies are im-

portant in the process of evaluating and choosing one of them. This is because, perceptions influence people's judgements, emotions, and decisions. Studying perceptions support their characterization as well as the development of design strategies to anticipate inconveniences in their implementation. This way, negative attitudes, fears and change resistance could be diminished.

In this paper, we present the results of research that was conducted in a state university setting located in Northeast México. Our purpose was to deepen in the understanding of the use of Pair Programming in university courses through experiences of students who implement it. Our research questions were: RQ1- What are the Pair Programming acceptance and assessment levels of students at this university? RQ2- What are the differences in Pair Programming acceptance and assessment in relation to student's previous programming experience, gender, and programming enjoyment? This study's objective is twofold: first, to characterize the students' acceptance and assessment of Pair Programming; also, to determine whether there are differences in these perceptions due to previous experience in pre-university courses, gender, and programming enjoyment. Our hypotheses were, Ha0: Students will report a high level of Pair Programming acceptance and assessment. Ha1: Students with prior programming experience will have a better perception of Pair Programming. Ha2: Women will have a better perception of Pair Programming than men. Ha3: Students with more enjoyment during programming will have a better perception of Pair Programming. The rest of the paper presents the methodology we followed in this research, the results we obtained after the analyses and a discussion that provides interpretation and insights on the results.

1.1 Literature review

In literature review, we covered relevant reported experiences of adopting Pair Programming in education and then, we highlighted how acceptance and assessment in Pair Programming have been measured. We realized that Pair Programming has proved to be an effective approach to promote learning. On the other hand, questionnaires have been extensively used to measure pair programming acceptance and assessment. Nevertheless, no thorough instrument validation procedures were found.

Pair programming in education: Positive aspects of working in pairs were identified by De Oliveira and Reboucas [16]. Students developed skills to enhance collaborative work and improve human interaction and communication. However, results also showed some disagreements and delays between participants which occurred when students were not compatible or had different expertise levels. Karthiekheyan, Ahmed and Jayalakshmi [17] focused on graduate students and found that working in pairs helped developing computer programming because it also promoted knowledge sharing and collaborative skills development.

Aarne, Peltola, Leinonen and Hellas [18] explored Pair Programming through students' attendance, expertise, and gender in introductory courses. Their results indicated that gender and previous programming experience correlate with students' participation. They also found that the reported enjoyment when working in pairs was not different in the assessed groups. Saltz and Shamshurin [19] found that paired working

could help in other contexts different from traditional software development. They implemented pair programming in a data science and big data graduate course with a positive outcome. Also, pair programming was implemented in a robot-control university course [20]. In that study, they noticed that students were able to share their knowledge despite their low or high skill level. It happened when the instructor organized them in pair work according to their social and programming skills instead of their grades.

Umapathy and Ritzhaupt [3] conducted a meta-analysis on Pair Programming which included quantitative studies in educational settings published between 2000 and 2014. They found that when it is appropriately implemented, Pair Programming has a positive influence on grades, exams performance and student persistence. These researchers suggested that it is essential to watch the role-playing in each team and to follow Pair Programming best practices during its implementation. Celepkolu and Boyer [13] highlighted that hybrid programming is different from Pair Programming because hybrid programming employs two people simultaneously working on one computer each. They compared both approaches in college beginner students and found that Pair Programming produces higher benefits than hybrid programming, such as real collaboration.

On the other hand, Bowman, Jarratt, Culver, and Segre [21] found that participants' previous programming experience positively relates to concepts understanding and confidence in developing software when working in pairs. Additionally, Choi [22] studied gender combinations in programmers' pairs. Compatibility and communication were different between same-gender and mixed pairs; however, no differences were found regard to coding. In a study of Pair Programming perceptions between men and women [23], it was found that both genders had favourable opinions, but women expressed fewer frustration feelings, increasing confidence, and making more friends. Women also admitted that learning was more accessible when working in pairs. As a conclusion, these authors recommended Pair Programming to eliminate obstacles that prevent women from participating in the software development field.

Moreover, Bowman, Jarratt, Culver, and Segre [21] emphasized on the importance of the gender factor in Pair Programming studies. They found that a female partner has positive effects on the results of Pair Programming regardless of the other partner's gender. These authors agree with [23]. They also perceive that Pair Programming is a way of including women in computer science.

Measuring acceptance and assessment of pair programming: In the literature, students' perceptions about Pair Programming are commonly studied through questionnaires. In some cases, previously designed or validated questionnaires have been used. Such is the case of Umar and Hui [24] who investigated the influence of learning styles and Pair Programming on job performance of Yang, Lee, and Chang [25] who related motivation to learn and retention to Pair Programming in data structure courses and of Aarne, Peltola, Leinonen and Hel-las [18] who studied enjoyment and attendance in Pair Programming. In other works, questionnaires have been validated through the factor analysis technique as part of the research process. For example: Choi [22], investigated the different gender combinations in programmer couples; Ghobadi, Campbell and Clegg [26] delved into pairings and knowledge transfer; and

Chen and Rea [8] investigated the relevance of Pair Programming approaches. Other articles reported the use of questionnaires and included with only some measures of consistency such as Cronbach's alpha. For example: Zhong, Wang and Chen [27] who studied the impact of social factors on Pair Programming outcomes in an elementary school. Zhong, Wang, Chen, and Li [28] who focused on the length of periods of change in Pair Programming roles. Some papers also included questionnaires in their methodology; however, they do not provided evidence of their reliability and validity. For example: Saltz and Shamshurin [19] who used Pair Programming in the data science field and Aottiwerch and Kokaew [29] who analysed the process of pairing students.

2 Materials and Methods

2.1 Participants

In this research, we examined 80 students enrolled in an undergraduate introductory programming class (Basic Programming) at a Mexican Public University. All of them performed a computer programming practice through Pair Programming in a single scheduled session. They had the previous individual experience, but none of them had programmed in pairs. They did not know the guidelines of this software development technique. Data was collected from three semesters in which students worked in Visual Basic.Net in a Windows-Forms project involving Database access without object-orientation. Table 1 shows the number of participants.

Table 1. Participants in this research

Data collection period	Participants
Fall, 2016	24
Fall, 2017	24
Fall, 2018	32

2.2 Procedure

In each academic period, the research was conducted in a single two-hour session. At the beginning of the session, the instructor explained the Pair Programming guidelines and software requirements. Then, students were randomly grouped in pairs; the attendance roll and random numbers were used for this purpose. Students were not allowed to switch pairs. A single computer was assigned to each pair of participants and software requirements were explained. The participants were asked to design and code a GUI (Graphical User's Interface) for database queries based on the user's requests. The Database Management System was Microsoft Access. The software was developed in a 70-minute time window and student roles were changed every five minutes. Students were not allowed to copy from other pairs and talk to other people. However, they could review their class notes. At the end of the session, students answered a questionnaire about their perception of the Pair Programming approach.

2.3 Data collection instrument

The questionnaire had six statements that were answered using a five-point Likert scale (see Table 2).

Table 2. The questionnaire used in this research.

Factor/Dimension	Items
Pair Programming Acceptance	I would like to program in pairs again. I liked working in pairs. I am satisfied with the way I programmed in pairs.
Pair Programming Assessment	Pair Programming reduces software development time. Software's quality is better with Pair Programming than with Solo Programming. Programmers' confidence is higher with Pair Programming than with individual Programming.

The instrument proved internal consistency, convergent validity and discriminant validity through exploratory and confirmatory factor analysis which were conducted in SPSS v25 and AMOS v24. The full validation process of the instrument is described in [30]. Two latent factors were found through exploratory factor analysis in SPSS: acceptance and assessment. Confirmatory factor analysis was performed afterwards in AMOS. The model fit values were: CMIN/DF = 1.15, CFI = 0.98, SRMR = 0.08, RMSEA = 0.06, PClose = 0.39. In Table 3, it can be seen that internal consistency was established with these values: Acceptance: CR = 0.82; Assessment: CR = 0.73. The convergent validity could be determined by the following values: Acceptance: CR = 0.82, AVE = 0.61; Assessment: CR = 0.73, AVE = 0.49. Finally, the discriminant validity was determined by the value of HTMT = 0.72.

Table 3. Internal consistency, convergent validity, discriminant validity for both factors in the questionnaire [30].

Factors	Cronbach's Alpha	CR	AVE	MSV	MaxR (H)	1. Pair Programming Acceptance	2. Pair Programming Assessment
1. Pair Programming Acceptance	0.767	0.82	0.61	0.34	0.91	0.78	
2. Pair Programming Assessment	0.704	0.73	0.49	0.34	0.80	0.58	0.69

2.4 Data analysis

For the statistical analysis we used SPSS 25, where an aggregate scale was calculated for each factor. These new values were used to perform descriptive and comparative analysis in each factor as shown in Table 4. A confidence level of 95% was used in the hypotheses testing and records with missing values were left out of the analyses. Two questionnaires -one from Fall 2016, and the other one from Fall 2018- were discarded because most of the answers were confusing or left blank.

Table 4. Statistical analyses conducted in each of the two factors.

Analyses	Studied groups	Statistical approach
Descriptive analysis	1. Two factors in the questionnaire. All participants included.	Mean, Standard Deviation, Median, interquartile range, KS, and SW normality tests.
Comparison of data collection periods.	Fall, 2016 (n=21) Fall, 2017 (n=24) Fall, 2018 (n=31)	Kruskal-Wallis Tests
Comparison between students with and without previous university programming experience.	1. Students who took programming courses in high school (n=33) 2. Students who did not take programming courses in high school (n=42)	Mann-Whitney Tests
Comparison between men and women.	Men (n=50) Women (n= 26)	Mann-Whitney Tests
Comparison among students with a low, medium, and high level of programming enjoyment.	Low level (n=26). Seven points or less. Medium level (n=27). Eight and nine points High level (n=23). Exactly 10 points.	Kruskal-Wallis Tests and Jonckheere-Terpstra Tests

Several answers can be arithmetically added to form a composite scale which represents a hypothetical construct [14]. Nevertheless, this process should be performed only with elements that prove to be reliable and valid. In regard to the statistical tests performed in this work, Mann-Whitney test is used to determine if two independent samples were obtained from the same population or two different populations with equal distribution [31]. Kruskal-Wallis test is used when it comes to comparing more than two groups. This test defines the research hypothesis that at least one of the groups is different from the rest [32]. Jonckheere-Terpstra test is useful to determine whether a statistically significant monotonic trend between an ordinal independent variable and a continuous or ordinal dependent variable exists [33]. This test is more powerful than the Kruskal-Wallis alternative because it takes advantage of the ordinal nature of the variable [34]. All of these are non-parametric tests which means they can be performed when normal distributions are not present in the data being analysed.

2.5 Ethical statement

All students gave their informed consent for inclusion in this research before their participation. The protocol was approved by Autonomous University of Tamaulipas.

3 Results

3.1 Descriptive analysis

The results of the descriptive analysis are presented in Table 5. Normal distribution was not observed in each factor; the significance value was close to 0 in Kolmogorov-

Smirnov and Shapiro-Wilk tests. The distributions for both factors can be seen in Figure 1 and Figure 2.

Table 5. Descriptive Analysis of both factors.

Factors	Maximum Value	Mean (Std. dev.)	Median	Interquartile Range (IQR)	Mode	Sig. KS Test	Sig.SW Test
Pair Programming Acceptance	15	12.44 (2.68)	13.00	4.00	15	0.0	0.0
Pair Programming Assessment	15	11.46 (2.29)	12.00	3.00	11,12	0.0	0.0

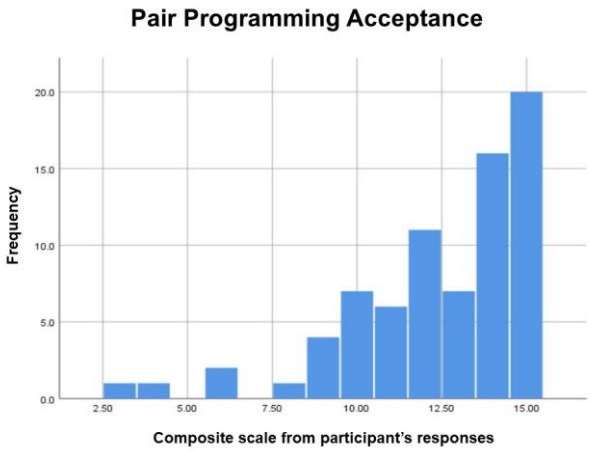

Fig. 1. Histogram for Pair Programming Acceptance.

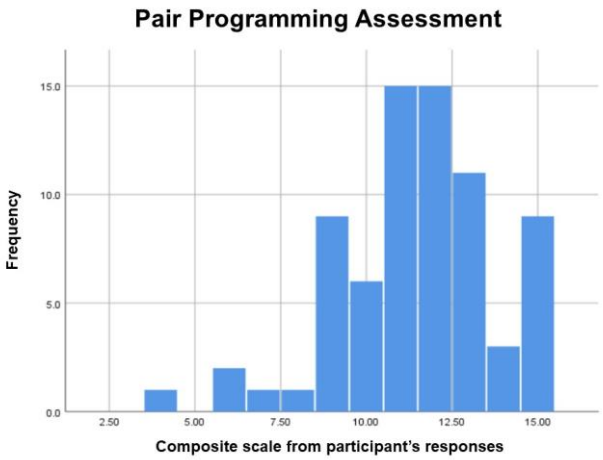

Fig. 2. Histogram for Pair Programming Assessment.

3.2 Comparative analysis

Comparison of data collection periods. We analysed each data collection from fall 2016, fall 2017 and fall 2018 to determine whether differences exist among these three groups. No differences were expected to be found. Descriptive statistics for this analysis are presented in Table 6.

Table 6. Descriptive statistics for three data collection periods. Factor 1: Pair Programming Acceptance. Factor 2: Pair Programming Assessment. (IQR=Interquartile Range)

Factor	Fall, 2016 (n=21)		Fall, 2017 (n=24)		Fall, 2018 (n=31)	
	Mean (Std. Dev.)	***Median (IQR)***	***Mean (Std. Dev.)***	***Median (IQR)***	***Mean (Std. Dev.)***	***Median (IQR)***
1	12.85 *(1.95)*	13.00 *(4.00)*	13.10 *(2.30)*	14.00 *(3.00)*	11.64 *(3.18)*	12.00 *(4.00)*
2	11.50 *(1.79)*	12.00 *(3.25)*	11.58 *(2.50)*	12.00 *(2.00)*	11.35 *(2.45)*	11.00 *(3.00)*

According to the results of the Kruskal-Wallis tests (Acceptance: H=3.74, d.f.= 2 p=0.15; Assessment: H=0.79, d.f. =2, p =0.67), no evidence was found to determine statistically significant differences in the data collected during the three periods. It was considered as a good indicator of homogeneity in the answers provided by the students.

Comparison between students with and without pre-university programming experience: We analysed two groups: 33 students with previous programming training before university and 42 students without prior programming experience. Table 7 presents the descriptive statistics in this analysis.

Table 7. Descriptive Statistics for students who took pre-university programming courses and those who did not.

Factor	Students who took pre-university programming courses (n=33)		Students who did not take pre-university programming courses (n=42)	
	Mean (Std. Dev.)	***Median (IQR)***	***Mean (Std. Dev.)***	***Median (IQR)***
Pair Programming Acceptance	12.78 *(1.93)*	13.00 *(2.50)*	12.23 *(3.15)*	14.00 *(4.25)*
Pair Programming Assessment	11.53 *(1.90)*	11.50 *(1.50)*	11.40 *(2.58)*	12.00 *(4.00)*

Results of the Mann-Whitney tests to determine differences in answers of students who took pre-university programming courses and those who didn´t (Acceptance: U=682.0, W=1585.0, Z=-0.11, p= 0.90; Assessment: U=622.50, W= 1087.50, Z= -0.87, p=0.93) show that no statistically significant differences can be established.

Comparison by gender: Two groups were configured for this analysis. One for men (n=50) and the other one for women (n=26). Descriptive statistics are presented in Table 8.

Table 8. Descriptive Statistics that characterize responses provided by men and women.

Factor	Women (n=26)		Men (n=50)	
	Mean (Std. Dev.)	***Median (IQR)***	***Mean (Std. Dev.)***	***Median (IQR)***
Pair Programming Acceptance	12.76 *(2.02)*	13.00 *(4.00)*	12.28 *(2.96)*	14.00 *(3.25)*
Pair Programming Assessment	12.04 *(1.89)*	12.00 *(2.00)*	11.24 *(2.43)*	12.00 *(3.00)*

The results from the Mann-Whitney tests (Acceptance: U=616.50, W=1891.50, Z=-0.37, p=0.70; Assessment: U=468.00, W=1693.00, Z=-1.16, p=0.24) show no evidence to determine statistically significant differences between the answers provided by men and women.

Comparison by programming enjoyment: In this analysis, we considered three groups: those who reported a high level of programming enjoyment (exactly 10 points), those with a medium level (8 or 9 points) and those with a low level of programming enjoyment (7 or fewer points). Descriptive Statistics of these three groups are presented in Table 9.

Table 9. Descriptive statistics of the three groups of students with different levels of programming enjoyment. Factor 1: Pair Programming Acceptance. Factor 2: Pair Programming Assessment.

Factor	Low level of programming enjoyment (n=26)		Medium level of programming enjoyment (n=27)		High level of programming enjoyment (n=23)	
	Mean (Std. Dev.)	***Median (IQR)***	***Mean (Std. Dev.)***	***Median (IQR)***	***Mean (Std. Dev.)***	***Median (IQR)***
1	11.46 *(2.85)*	12.00 *(4.25)*	12.25 *(2.90)*	13.00 *(3.00)*	13.78 *(1.4)*	14.00 *(2.00)*
2	11.38 *(2.02)*	11.00 *(2.25)*	11.50 *(2.48)*	12.00 *(3.75)*	11.66 *(2.5)*	12.00 *(3.00)*

In the results of the Kruskal-Wallis tests to seek differences between students with high and low levels of programming enjoyment (Acceptance: H=10.34, d.f.=2, p=0.00; Assessment: H=0.567, d.f.=2, p=0.75), statistically significant differences were found. Students with a high level of programming enjoyment reported the most top scores of Pair Programming acceptance (mean rank = 49.85). In contrast, those students with a low level of programming enjoyment reported the lowest scores (mean rank = 30.06).

Jonckheere-Terpstra tests were conducted afterwards. The stated hypothesis was that there would be a positive monotonic trend in both "Pair Programming Acceptance" and "Pair Programming Assessment" factors as the programming enjoyment increased. In the results of these tests, a statistically significant increasing monotonic trend is evident in Pair Programming acceptance (standardized statistic = 3.20, p = 0.00). The Kendall's τb between Pair Programming acceptance and programming enjoyment was 0.30, p=0.00. No differences were found in Pair Programming assessment (standardized statistic=0.72, p=0.47).

4 Discussion

4.1 Revisiting the research questions and stated hypotheses

A summary of the research questions, stated hypotheses, their results, conclusions, and explanations are presented in Table 10.

Table 10. Summary of research questions, stated hypotheses, results and conclusions.

RQ1.- What are the Pair Programming acceptance and assessment levels of students at this university?			
Research Hypotheses	***Results***	***Conclusions***	***Explanation***
H_{a0}: Students will report a high level of Pair Programming acceptance and assessment	Histograms for Acceptance and Assessment show that the highest frequencies are concentrated in the highest values of the scales.	H_{a0} is accepted	Most of the students reported high values of Pair Programming Acceptance and Assessment.
RQ2.- What are the differences in pair programming acceptance and assessment in relation to student's previous programming experience, gender, and programming enjoyment?			
Research Hypotheses	***Results***	***Conclusions***	***Explanation***
H_{a1}: Students with prior programming experience will have a better perception of pair programming.	Mann Whitney Tests Acceptance: p=0.90 Assessment: p=0.93	There is not enough evidence to support H_{a1}.	No differences could be established in acceptance and assessment between students with and without prior programming experience.
H_{a2}: Women will have a better perception of pair programming than men.	Mann-Whitney Tests Acceptance: p=0.70 Assessment: p=0.24	There is not enough evidence to support H_{a2}.	No differences could be established in acceptance and assessment between men and women.
H_{a3}: Students with more programming enjoyment will have a better perception of pair programming.	Kruskal-Wallis Tests Acceptance: p=0.00 (Students with a high level of programming enjoyment: median=14. Students with a low level of enjoyment: median=12). Assessment: p=0.75 Jonckheere-Terpstra tests Acceptance: $p=0.00$, standardized statistic=3.20. Assessment: $p=0.47$	H_{a3} is accepted. (Hypothesis was accepted for "Pair Programming acceptance")	Students who enjoy programming more report higher levels of Pair Programming acceptance.

4.2 Interpretation and implication of results

The results showed that Pair Programming was positively accepted and rated by students. Pair Programming acceptance increased as the programming enjoyment increased. No differences were found in the other comparisons performed by gender and previous programming experience. These findings provide evidence on the use of Pair Programming in the Mexican University classrooms and encourage its adoption

in programming courses. However, more research is needed to identify the best strategies to implement this approach with a high level of acceptance and effectivity among the students. It is also relevant to determine how often it should be used in classes. Another aspect that needs further research is the partner´s influence on the perceptions and effectivity of Pair Programming.

4.3 Comparison of the results and the analysed literature

Our results agreed with [18] because we also found no significant differences in the perception of Pair Programming due to previous programming experience and gender of participants. On the other hand, we agreed with [3] because we also conclude that if implemented correctly, Pair Programming can lead to positive results for students. Finally, like [23], we found that men and women assessed Pair Programming positively. However, we do not have enough elements to conclude that women have leaned towards some other aspects.

4.4 Contribution

Research on Pair Programming in university as a tool to help students successfully complete their professional studies is relevant to the education field and to the community development. As part of institutional strategies, pair programming would reduce failing and dropping out of higher education which would have a great positive impact on our community.

4.5 Limitations

Even though the study included participants from three different cohorts, all the students belonged to a single university and the same undergraduate program. On the other hand, the research focused only on the acceptance and assessment of Pair Programming according to the students' perceptions and their previous experience, gender, and level of programming enjoyment.

4.6 Challenges

The study was conducted at a Mexican state university where an average of 200 students spread over nine academic periods were enrolled in its IT undergraduate program. The "Basic Programming" course was offered only once a year and the number of students taking it was few. Therefore, for this study we had to work with the total available students in each academic period. On the other hand, we were limited by the two-hour long sessions that were scheduled for this class. The students were not able to stay after the session ended because they had to attend a different course. That is why the time had to be organized in advance and well administered at the time of the Pair Programming exercise.

5 Conclusion

Pair Programming is an agile technique to develop high-quality working software in a short time. This approach has proved to be helpful in educational and business settings because it promotes collaborative work, knowledge transfer and skills development among participants. Nowadays, researching Pair Programming is not a novel or a completed activity. Even though there is a wide variety of related literature, there are still lots of relevant aspects that need further research to understand them better. It is due to the complex human nature of the participants.

Our study revealed that Pair Programming was positively accepted and assessed by university students enrolled in basic programming courses. We also found an increasing trend in its acceptance as the programming enjoyment increased. No other differences were found in the comparisons performed according to gender and previous programming experience. While we found Pair Programming is helpful and easy-to-implement in university courses and students positively accepted and assessed this learning approach. Therefore, based on our experience, we recommend Pair Programming adoption in programming university courses as part of a broader strategy to involve students more in the effective learning of programming. Nevertheless, we are aware that our results are limited, and more research is needed to determine some other important facets of Pair Programming.

As future work for us and for other researchers, we recommend going over these results from different perspectives. For instance, studying the optimum frequency for its classroom/lab implementation, the partners, their backgrounds, and their dynamics in the learning process. Longitudinal studies would also be of great value to nourish the available knowledge on Pair Programming. We also think that qualitative approaches would broaden our findings and derive new lines of research.

We know that pair work is not a solution to every problem found in the learning and teaching of programming. Nonetheless, we believe it is a helpful tool to engage students in programming along with their professional studies.

6 Acknowledgement

Authors would like to thank Autonomous University of Tamaulipas, México and Autonomous University of Nuevo León for their support while conducting this research.

7 References

[1] R. Shen, D. Y. Wohn, and M. J. Lee, “Programming learners’ perceptions of interactive computer tutors and human teachers,” *Int. J. Emerg. Technol. Learn.*, vol. 15, no. 9, pp. 123–142, 2020, https://doi.org/10.3991/ijet.v15i09.12445.

[2] S. I. Malik, M. Shakir, A. Eldow, and M. W. Ashfaque, “Promoting algorithmic thinking in an introductory programming course,” *Int. J. Emerg. Technol. Learn.*, vol. 14, no. 1, pp. 84–94, 2019, https://doi.org/10.3991/ijet.v14i01.9061.

[3] K. Umapathy and A. D. Ritzhaupt, "A meta-analysis of pair-programming in computer programming courses: Implications for educational practice," *ACM Trans. Comput. Educ.*, vol. 17, no. 4, pp. 1–13, 2017, https://doi.org/10.1145/2996201.

[4] T. Koulouri, S. Lauria, and R. D. Macredie, "Teaching introductory programming: A quantitative evaluation of different approaches," *ACM Trans. Comput. Educ.*, vol. 14, no. 4, 2014, https://doi.org/10.1145/2662412.

[5] S. Papadakis, "Is pair programming more effective than solo programming for secondary education novice programmers? A case study," *Int. J. Web-Based Learn. Teach. Technol.*, vol. 13, no. 1, pp. 1–16, 2018, https://doi.org/10.4018/ijwltt.2018010101.

[6] S. Papadakis, "Robots and Robotics Kits for Early Childhood and First School Age," *Int. J. Interact. Mob. Technol.*, vol. 14, no. 18, pp. 34–56, 2020, https://doi.org/10.3991/ijim.v14i18.166311.

[7] R. Poonam and C. M. Yasser, "An experimental study to investigate personality traits on pair programming efficiency in extreme programming," *2018 5th Int. Conf. Ind. Eng. Appl. ICIEA 2018*, pp. 95–99, 2018, https://doi.org/10.1109/iea.2018.8387077.

[8] K. Chen and A. Rea, "Do Pair Programming Approaches Transcend Coding? Measuring Agile Attitudes in Diverse Information Systems Courses," *J. Inf. Syst. Educ.*, vol. 29, no. 2, pp. 53–64, 2018.

[9] M. Villamor and M. M. Rodrigo, "Predicting successful collaboration in a pair programming eye tracking experiment," *UMAP 2018 - Adjun. Publ. 26th Conf. User Model. Adapt. Pers.*, pp. 263–268, 2018, https://doi.org/10.1145/3213586.3225234.

[10] R. V. Roque Hernández, J. Cárdenas Domínguez, A. López Mendoza, J. A. Herrera Izaguirre, and C. M. Juárez Ibarra, "Comparison of Pair and Solo Programming through software metrics in University Students' Projects," *RIDE Rev. Iberoam. para la Investig. y el Desarro. Educ.*, vol. 10, no. 19, 2019, https://doi.org/10.23913/ride.v10i19.549.

[11] Z. Nurbekova, V. Grinshkun, G. Aimicheva, B. Nurbekov, and K. Tuenbaeva, "Project-based learning approach for teaching mobile application development using visualization technology," *Int. J. Emerg. Technol. Learn.*, vol. 15, no. 8, pp. 130–143, 2020, https://doi.org/10.3991/ijet.v15i08.12335.

[12] M. O. Smith, A. Giugliano, and A. Deorio, "Long Term Effects of Pair Programming," *IEEE Trans. Educ.*, vol. 61, no. 3, pp. 187–194, 2018, https://doi.org/10.1109/te.2017.2773024.

[13] M. Celepkolu and K. E. Boyer, "The importance of producing shared code through pair programming," *SIGCSE 2018 - Proc. 49th ACM Tech. Symp. Comput. Sci. Educ.*, vol. 2018-Janua, pp. 765–770, 2018, https://doi.org/10.1145/3159450.3159506.

[14] W. Zikmund, B. Babin, J. Carr, and M. Griffin, *Business Research Methods*, 9th ed. Moson, OH, EU: Cengage Learning, 2013.

[15] W. Sun, G. Marakas, and M. Aguirre-Urreta, "The Effectiveness of Pair Programming," *IEEE Softw.*, pp. 72–79, 2016. https://doi.org/10.1109/ms.2015.106

[16] T. A. N. de Oliveira and A. D. Reboucas, "The Use of Pair Programming to Support Introductory Programming Teaching: A Qualitative Study," pp. 65–68, 2018, doi: 10.1109/laclo.2018.00025. https://doi.org/10.1109/laclo.2018.00025

[17] K. Karthiekheyan, I. Ahmed, and J. Jayalakshmi, "Pair Programming for Software Engineering Education: An Empirical Study," 2018.

[18] O. Aarne, P. Peltola, J. Leinonen, and A. Hellas, "A study of pair programming enjoyment and atendance using study motivation and strategy metrics," *SIGCSE 2018 - Proc. 49th ACM Tech. Symp. Comput. Sci. Educ.*, vol. 2018-Janua, pp. 759–764, 2018, https://doi.org/10.1145/3159450.3159493.

[19] J. S. Saltz and I. Shamshurin, "Does pair programming work in a data science context? An initial case study," in *Proceedings - 2017 IEEE International Conference on Big Data, Big Data 2017*, 2017, pp. 2348–2354, https://doi.org/10.1109/bigdata.2017.8258189.

[20] M. Ueda, Y. Suzuki, A. Itai, T. Yamashita, and N. Ishii, "Report on Behavior Analysis Using Text Mining in Pair Programming Education," *Proc. - 2017 6th IIAI Int. Congr. Adv. Appl. Informatics, IIAI-AAI 2017*, pp. 1017–1018, 2017, https://doi.org/10.1109/iiai-aai.2017.154.

[21] N. A. Bowman, L. Jarratt, K. C. Culver, and A. M. Segre, "How prior programming experience affects students' pair programming experiences and outcomes," *Annu. Conf. Innov. Technol. Comput. Sci. Educ. ITiCSE*, pp. 170–175, 2019, https://doi.org/10.1145/3304221.3319781.

[22] K. S. Choi, "A comparative analysis of different gender pair combinations in pair programming," *Behav. Inf. Technol.*, vol. 34, no. 8, pp. 825–837, 2015, https://doi.org/10.1080/0144929x.2014.937460.

[23] K. M. Ying, L. G. Pezzullo, M. Ahmed, K. Crompton, J. Blanchard, and K. E. Boyer, "In Their Own Words: Gender Differences in Student Perceptions of Pair Program-ming," 2018, doi: 10.1145/3287324.3287380.

[24] I. N. Umar and T. H. Hui, "Learning Style, Metaphor and Pair Programming: Do they Influence Performance?," *Procedia - Soc. Behav. Sci.*, vol. 46, pp. 5603–5609, 2012, https://doi.org/10.1016/j.sbspro.2012.06.482.

[25] Y. F. Yang, C. I. Lee, and C. K. Chang, "Learning motivation and retention effects of pair programming in data structures courses," *Educ. Inf.*, vol. 32, no. 3, pp. 249–267, 2016, https://doi.org/10.3233/efi-160976.

[26] S. Ghobadi, J. Campbell, and S. Clegg, "Pair programming teams and high-quality knowledge sharing: A comparative study of coopetitive reward structures," *Inf. Syst. Front.*, vol. 19, no. 2, pp. 397–409, 2017, https://doi.org/10.1007/s10796-015-9603-0.

[27] B. Zhong, Q. Wang, and J. Chen, "The impact of social factors on pair programming in a primary school," *Comput. Human Behav.*, vol. 64, pp. 423–431, 2016, https://doi.org/10.1016/j.chb.2016.07.017.

[28] B. Zhong, Q. Wang, J. Chen, and Y. Li, "Investigating the period of switching roles in pair programming in a primary school," *Educ. Technol. Soc.*, vol. 20, no. 3, pp. 220–233, 2017.

[29] N. Aottiwerch and U. Kokaew, "The analysis of matching learners in pair programming using K-means," *2018 5th Int. Conf. Ind. Eng. Appl. ICIEA 2018*, pp. 362–366, 2018, https://doi.org/10.1109/iea.2018.8387125.

[30] R. V. Roque-Hernández, "Diseño de un instrumento para medir la aceptación y los beneficios percibidos de la programación por pares en los cursos universitarios," *Acta Univ.*, vol. 30, pp. 1–12, 2020, https://doi.org/10.15174/au.2020.2877.

[31] R. Levin and D. Rubin, *Estadística para administración y economía*, 7th ed. México: Pearson, 2010.

[32] S. Siegel and N. J. Castellán, *Estadística no paramétrica aplicada a las ciencias de la conducta*, 2nd ed. México: Trillas, 2015. https://doi.org/10.26439/persona1998.n001.1715

[33] L. Statistics, "Laerd Statistics," 2020. https://statistics.laerd.com/.

[34] M. Kraska-Miller, *NonParametric Statistics for Social and Behaviorall Sciences*, 1st ed. New York, NY, USA: CRC Press, 2014.

8 Authors

Ramón Ventura Roque-Hernández is currently a Professor at Faculty of Commerce, Administration and Social Sciences. Autonomous University of Tamaulipas. Ayuntamiento Sur SN. Col. Infonavit Fundadores. Post Code 88000. Nuevo Laredo, Tamaulipas, México. Email: rvhernandez@uat.edu.mx

Sergio Armando Guerra-Moya is currently a Professor at Faculty of Public Accounting and Administration. Autonomous University of Nuevo León. Av. Universidad s/n. Ciudad Universidad. Post Code 66455. San Nicolás de los Garza, Nuevo León, México. Email: sagm52@hotmail.com

Frida Carmina Caballero-Rico is currently a Professor at Center of Excellence. Autonomous University of Tamaulipas, México. Adolfo López Mateos University Center. Post Code 87149. Ciudad Victoria, Tamaulipas, México. Email: fcaballer@uat.edu.mx

Article submitted 2020-09-28. Resubmitted 2021-01-16. Final acceptance 2021-01-17. Final version published as submitted by the authors.