Effect of Knowledge-Based Team Composition on Effectiveness of Pair Programming

N. D. Gunasekara

(Lecturer in IT) Department of Information Technology ATI Gampaha, SLIATE, Sri Lanka

DOI: https://dx.doi.org/10.47772/IJRISS.2024.802097

Received: 29 January 2024 ; Revised: 08 February 2024; Accepted: 12 February 2024; Published: 15 March 2024

ABSTRACT

Over years variety of tools has been used in the class room to support student learning. Among them, pair programming is trending as a pedagogical tool for programming courses in higher education. Implementation of pair programming is inadequate, choosing the right factors for successful adoption of this method is vital. This study aims at investigating the effect of knowledge-based team composition on effectiveness of pair programming. First year students following HNDIT at ATI Gampaha in the academic year 2020 had participated for the experiment. Data was collected using four instruments. The final exam scores, assignment scores, questionnaire, code quality. Mann Whitney U test is used to analyze assignment score and exam score and the code quality was determined by the defect density. The results revealed that students who had paired performed better in the assignments and they had graded better in the final exam as well. Knowledge based pair programming has positively affected code quality also. Further it reports willingness among students to use pair programming in the academic field when they are paired in a compatible way. Thus, it can be concluded that knowledge based team composition has positively affected to pair programming. These results are consistent with those of other studies that have done on the same topic.

Keywords: Pair Programming, Knowledge Based, Team Composition, Effectiveness, Code Quality

INTRODUCTION

Over years variety of tools has been used in the class room to support student learning. Among them, pair programming is trending as a pedagogical tool for programming courses in higher education. Pair programming is a type of agile software development technique where two programmers work together with one computer for developing required software. (Karthiekheyan, Ahmed, & Jayalakshmi, 2018). Pair programming is adhered to the education set up because of many positive points identified in the industrial environment such as product quality, team productivity, schedule adherence of tasks, learning (Lassenius, 2007) (Muller, 2003). Research reveals many pedagogical benefits of pair programming like low failure rates, low attrition rates (Corney, Teague, & Thomas, 2010), collaboration (Karthiekheyan, Ahmed, & Jayalakshmi, 2018), and enjoyment (Faja, 2014). Implementation of pair programming is inadequate, choosing the right factors for successful adoption of this method is vital. Critics argue that the average student learning in isolation performs significantly less well than those learning with collaboration and mediation. Students learn through talk, discussion, and argumentation.

Team effectiveness model which was introduced by Faja (Faja, 2014) presents team environment, team composition, team design and team process as the aspects which matters for effectiveness in pair programming. In previous studies various team composition methods have been implemented such as

matching pairs vs random pairs , same partner vs changing partners. Matching pairs were formed based on academic performance (Choi, 2009) (Zacharis, 2011), personality traits (Choi, 2009) and gender (Werner, 2004). This study aims at investigating the effect of knowledge-based team composition on effectiveness of pair programming.

Objectives of the Study

Over years variety of tools has been used in the class room to support student learning. Among them, pair programming is trending as a pedagogical tool for programming courses in higher education. Pair programming is a type of agile software development technique where two programmers work together with one computer for developing required software. (Karthiekheyan, Ahmed, & Jayalakshmi, 2018). Research reveals many pedagogical benefits of pair programming like low failure rates, low attrition rates (Corney, Teague, & Thomas, 2010), collaboration (Karthie kheyan, Ahmed, & Jayalakshmi, 2018), and enjoyment (Faja, 2014). Implementation of pair programming is inadequate, choosing the right factors for successful adoption of this method is vital. Team effectiveness model which was introduced by Faja (Faja, 2011) presents team environment, team composition, team design and team process as aspects which matters for effectiveness in pair programming. In previous studies various team composition methods have been implemented. This study aims at investigating the effect of knowledge-based team composition on effectiveness of pair programming.

In order to assist the aim, the study intends to achieve the following specific objectives:

- examine academic performance of pair programmers with solo programmers
- assess code quality of pair programmers with solo programmers
- compare lab session participation of pair programmers with solo programmers
- determine satisfaction of pair programming

LITERATURE REVIEW

In recent years pair programming has brought considerable attention in higher education and vast amount of research had been done related to pair programming in education spanning across different areas. When implementing pair programming, different pair formation techniques can be used such as matching pairs vs random pairs, same partner vs changing partners. In previous studies matching pairs were formed based on academic performance (Choi, 2009) (Zacharis, 2011), personality traits (Choi, 2009) and gender (Werner, 2004). When considering academic performance most have paired students with similar skills together (Choi, 2009) (Zacharis, 2011) (Cliburn, 2003) but there are rare situations where a weak student is paired with a smart one (Chigona & Pollock, 2008). But that may result in intra pair stress (Faja, 2011) and the best approach is to group similar skill students together (Cliburn, 2003) (Bevan, Werner, & McDowell, 2002)

Many studies have been conducted to measure the effectiveness of pair programming in the educational domain. When measuring the effectiveness different aspects such as students' perception, students' performance and work quality had been considered. In 2014 Faja had done perception survey-based analysis to determine the effectiveness of pair programming as a teaching tool. The study reveals that when comparing different effectiveness measures, students' perceived learning, quality of work, and enjoyment during pair programming was found to be at a higher level than increased productivity outcome. (Faja, 2014) In 2002 a research was done to assess the effectiveness and usability of using pair programming to improve programming language learning, productivity, and code quality. The results reported that students who programmed in pairs produced better programs, completed the course at higher rates, and performed about as well on the final exam as students who programmed independently. (McDowell, 2002). But as far as the researchers observed there is no research considering knowledge based pairing when evaluating

effectiveness of pair programming.

The researches have been done not only on evaluation of pair programming, but also in implementation of pair programming. Williams et al introduced nine guidelines to implement pair programming in class room and then revised it in 2008 to 11 guidelines (Williams, McCrickard, Layman, & Hussein, 2008). Bevan et al presents six guidelines for the use of pair programming in a freshman programming class (Bevan, Werner, & McDowell, 2002)

Conceptual Framework

Review of literature results in a conceptual framework shown in Figure 1. The modal hypothesize that the students' perception, students' performance and code quality gives an indication on the effectiveness of pair programming.

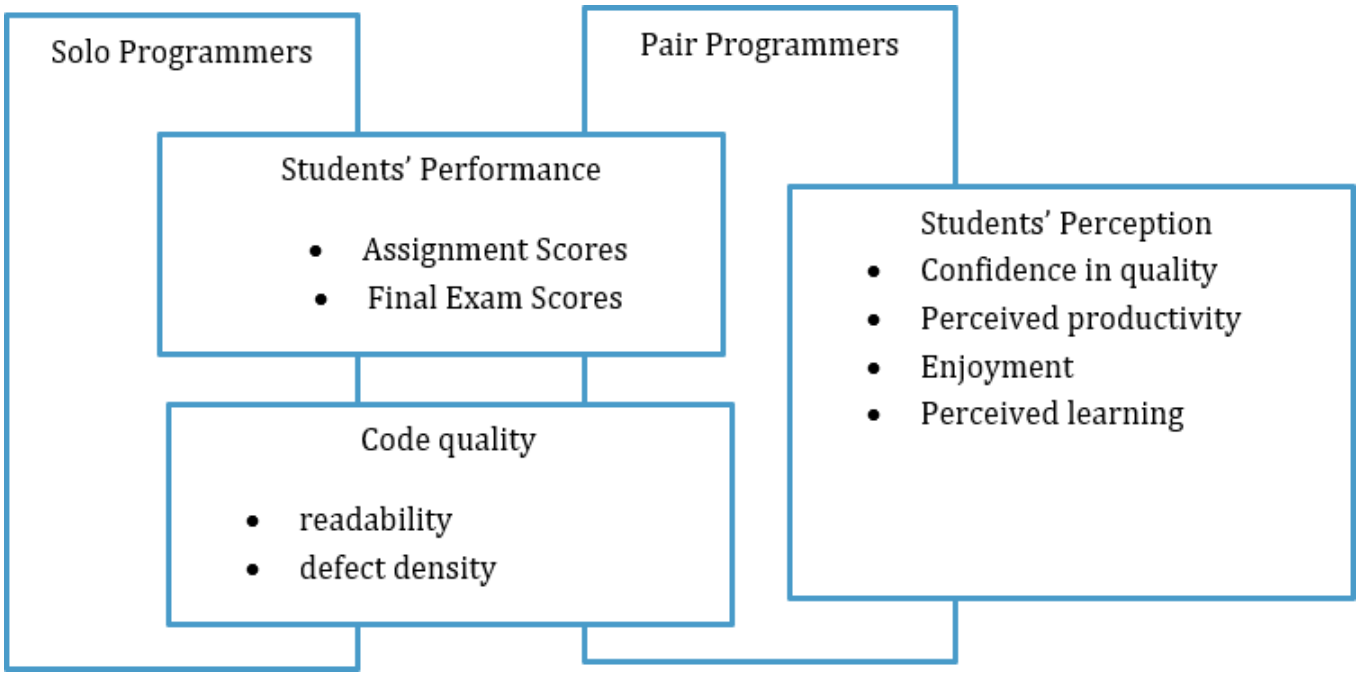

Figure 1:Conceptual Framework

Students' performance: The assignment is given individually for solo programmers and as a pair for pair programmers then scores will be compared. Final exam which is done individually, will be considered and compared to assess whether there is an improvement of pair programmers in their individual performance

Students' perception: Through four constructs it was tested whether the students' pair programming activities has positively related to their perception of pair programming effectiveness. Also it was tested whether perceived partner's effort has influenced students' perception of effectiveness in pair programming.

Code quality: code quality was measured considering the readability and defect density. Meaningful identifier naming, indentation, commenting is considered for readability and defect density depends on the number of erroneous code lines.

METHODOLOGY

The target population consist of first year students following HNDIT at ATI Gampaha in the academic year 2020. Their medium of study is English and had completed the first year of their diploma programme. Data

was collected using five instruments. The final exam scores, assignment scores, questionnaire, attendance, instructor's ratings. final examination results of Structured Programming subject which held in the previous semester was considered for student pairing in the experimental group. assignment scores of the pairing groups and non pairing students were considered. The individual final exam grading of both groups used for comparison. Questionnaire was used to test four variables perceived learning, confidence in quality, enjoyment and productivity. Attendance of controlled group and experimental group was compared to check the enthusiasm in participating for programming. Instructor's rating for the codes were considered to check the code quality.

The experimental procedure is designed based on the guidelines given by Williams et al (Williams, McCrickard, Layman, & Hussein, 2008)and Bevan et al (Bevan, Werner, & McDowell, 2002) in their research. Initially two groups were formed randomly as administration division divides students in to groups on enrolment sequence. Object Oriented Programming subject consisted of four hour laboratory activities weekly. From week 2 to week 10 laboratory activities were replaced by pair programming sessions for experimental group. The comparison group performed laboratory activities individually under supervision of the instructors. Same instructors guided both groups and the same activities and assignments were given for both groups.

Following steps has been executed with the experimental group.

In the first week the instructor explained pair programming to students using a presentation. They were informed about the attendance policy for the final exam and the lateness penalty. Students paired by the instructor based on the final examination results of Structured Programming subject which held in the previous semester and the students with equal or similar grades paired together. 23 pairs were constructed. From the second week onwards, students were given lab sheets which they had to complete as a pair. The instructor observed students throughout the lab session to make sure they are switching the roles of ‘’driver" and "navigator" and also are engaging in the activities. The students were evaluated based on both individual and collaborative work. Two in class assignments were given where one assignment is individual. The experimental group will had to complete the other assignment as a pair and the comparison group had complete it individually.

RESULTS

- Scores on programming assignment

Here students required to complete the programming assignment in pairs (N=16) while other students wrote the program independently (N=35). To compare whether programming scores differed as a function of pair-programming experience, analysis of variance (ANOVA) was conducted. Among all students who completed the course, students in the pairing group scored higher on the programming assignments (M=39.8%), than those in the non-pairing group(M=23%) $F(1, 53) = 5.049$, $p < .05$. it is highly unlikely that we would have obtained these results if pairing didn't actually influence the quality of the programs.

There are two possible explanations for this difference of means. First, pair-programming might have influenced the students to write better programs. A second possibility is that the mean assignment score in the pairing group was artificially inflated. Because both members of the pairs earned the same score on each of the programming assignments, overall scores in the group may have simply reflected the performance of the stronger student in each pair. In the most extreme case, it is possible that each of the pairs in the pairing section consisted of one partner in the top half and one partner in the bottom half of the class, resulting in a mean assignment score for the whole class that only represented the performance of the top 50%. If pair-programming did not improve the quality of the programming assignments, then the scores in the pairing class should have been approximately equal to the scores of the strongest 50% of students in the non-pairing

group. To test this, an ANOVA was performed to compare the assignment scores of all students in the pairing class to the students in the top half of the non-pairing group (student ranking was determined by final exam scores of structured programming subject in the previous semester). Overall, the scores from the entire pair-programming section (M=39.89%) were higher than the scores of the top half of the non-pairing group(M=32%), F (1,37) =6.08, p<.05. This suggests that the best 50 programs

from a group of 100 students working alone, would not be as good as the programs produced by 50 pairs of students. So, it appears that working as a team has improved the knowledge of writing programs.

- pair programming and final exam scores

H0: paired students got less grades in the final exam than students who worked individually.

H1: paired students got better grades in the final exam than students who worked individually.

Mann Whitney U test had been used to check the relationship between pairing and the students' grades. As per table 1 one tailed p-value is 0.099/2=0.049 since p<0.05 the null hypothesis is rejected. Thus the results indicate that paired students got better grades in the final exam than students who worked individually.

	Exam Result
Mann-Whitney U	551.000
Wilcoxon W	1146.000
Z	-1.649
Asymp. Sig. (2-tailed)	.099
a. Grouping Variable: paired	

Table 1: Test Statistics[a]

- Pair programming and attendance

H0: student attendance depends on pairing.

H1: student attendance does not depend on pairing

When considering the contingency table of pairing vs attendance, the significance value 0.212 > 0.05 which indicates that the rows and columns of the contingency table are independent. The students' attendance is independent of pairing according to the chi square test results of table 2. Many factors may contribute to the independence between variables. Majority of the students may find the subject interesting or may difficult. Some times students may find participating lectures help them to learn programming better. (Gunasekara, 2021)

	Value	df	Asymp. Sig. (2-sided)
Pearson Chi-Square	9.600[a]	7	.212
Likelihood Ratio	10.865	7	.145
Linear-by-Linear Association	.165	1	.685
N of Valid Cases	83		
a. 10 cells (62.5%) have expected count less than 5. The minimum expected count is .92.			

Table 2: Chi Square Tests

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS |Volume VIII Issue II February 2024

- pair programming and students' perception

outcome	mean	standard deviation
confidence in quality	2.999	6.95
perceived productivity	2.08	0.935
enjoyment	3.07	6.995
perceived learning	1.68	0.838

Table 3:Students' Perception

Results in the table 3 show that students are unable to make a statement about the quality of the program when they are pair programmed. They agree that they could finish the work quicker thus the work is productive when they write programs together.

- pair programming and code quality

The code quality was determined by the defect density. The researcher has gone through a set of programs written to solve a specific task and then defect density was calculated using the formula

Defect density = defects/ lines of code

H0: Defect density is same for paired and non paired students

H0: Defect density in non paired students is higher compared to paired students

The p value of Levene's test is 0.000 in table 4. Thus null hypothesis is rejected. so it is assumed variance of two groups are not equal. The p value<0.05 the null hypothesis is rejected and concluded that defect density in non paired students is higher compared to paired students.

Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
defect Density	Equal variances assumed	169.814	.000	-5.680	87	.000	-.572061982	.100708931	-.772231886	-.371892077
	Equal variances not assumed			-5.536	52.860	.000	-.572061982	.103334763	-.779338094	-.364785870

Table 4: Independent Sample Test

CONCLUSION

The objective of this study was to examine effectiveness of pair programming when the students were paired based on their knowledge. The conceptual framework considered students' performance, code quality and the students' perception to measure the effectiveness. The students' performance was measured by the parameters assignment score and final exam score. Students who had paired performed better in the assignments and they had graded better in the final exam as well. Knowledge based pair programming has positively affected code quality also. Further it reports willingness among students to use pair programming in the academic field when they are paired in a compatible way. This will lead to students' satisfaction and better academic performance. When considering the limitations of the study attrition bias may be included up to some extent as there is a 7% drop out. Further, a more comprehensive analysis could have been done if the study had been carried out for several programming subjects. The study was done considering first year students. The study could have been expanded to analyze effectiveness of students in higher academic grades as well, This study considered only one parameter 'knowledge' for team composition so there is room for improvement in studying other parameters such as personality type, gender for pair formation and it's relationship with effectiveness. However the results in this study are consistent with those of other studies that have done on the same topic.

REFERENCE

1. Alrawashdeh, T. (2017). Assessing the Effectiveness and Usability of Using Pair Programming to Improve Programming Language Learning , Productivity , and Code Quality. Journal of Theoretical and Applied Information Technology.
2. Bevan, J., Werner, L., & McDowell, C. (2002). Guidelines for the Use of Pair Programming in a Freshman Programming Class. Proceedings of the 15th Conference on Software Engineering Education and Training, (pp. 100-107).
3. Choi, K. D. (2009). Pair Dynamics in Team Collaboration. Computers in Human Behavior, 844-852.
4. Corney, M., Teague, D., & Thomas, R. (2010). Engaging students in Programming. 12th Australasian Computing Education Conference (ACE 2010) (pp. 63-72). Brisbane: Australian Computer Society, Inc.
5. Derus, S., & Ali, A. (2012). Difficulties in learning programming: Views of students. 1st International Conference on Current Issues in Education, (pp. 74-79).
6. Faja, S. (2011). Pair programming as a team based learning activity a review of research. Issues in information systems, 207-216.
7. Faja, S. (2014). Evaluating Effectiveness of Pair Programming as a Teaching Tool in Programming Courses. Information Systems Education Journal.
8. Gunasekara, N. (2021). Gender Differences in Learning Programming: An Analysis by Students' Perception. International Journal of Research and Innovation in Social Science, 149-155.
9. Karthiekheyan, K., Ahmed, I., & Jayalakshmi, J. (2018). Pair Programming for Software Engineering Education: An Empirical Study. The International Arab Journal of Information Technology, 246-255.
10. Lassenius, J. V. (2007). Perceived Effects of Pair Programming in an Industrial Context. 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (pp. 211-218). IEEE.
11. McDowell, C. W. (2002). The effects of pair-programming on performance in an introductory programming course. Proceedings of the 33rd SIGCSE technical symposium on Computer science education, (pp. 38-42).
12. Muller, F. P. (2003). Analyzing the cost and benefit of pair programming. 5th International Workshop on Enterprise Networking and Computing in Healthcare Industry (pp. 166-177). IEEE.
13. Neha Katira, L. W. (2004). On understanding compatibility of student pair programmers. 35th SIGCSE technical symposium on Computer science education. SIGCSE '04.

14. Neha Katira, L. W. (2005). Towards increasing the compatibility of student pair programmers. 27th International Conference on Software Engineering, 2005. ICSE 2005. IEEE.
15. Werner, L. H. (2004). Pair-Programming Helps Female Computer Science Students. ACM Journal of Educational Resources in Computing, 1-8.
16. Williams, L., Layman, L., Osborne, J., & Katira, N. (2006). Examining the compatibility of student pair programmers. AGILE 2006 (AGILE'06). Minneapolis, MN, USA: IEEE.
17. Williams, L., McCrickard, D. S., Layman, L., & Hussein, K. (2008). Eleven Guidelines for Implementing Pair Programming in the Classroom. Agile 2008 Conference. Toronto, ON, Canada: IEEE.
18. Zacharis, N. Z. (2011). Measuring the Effects of Virtual Pair Programming in an Introductory Programming. IEEE Transactions on Education, 168-170.