Jurnal Mantik, 5 (4) (2022) 2531-2535

Published by:Institute of Computer Science (IOCS)

Jurnal Mantik

Journal homepage: www.iocscience.org/ejournal/index.php/mantik/index

Strategies for an Effective Distributed Pair Programming

Mario Elyezer Subekti Simaremare

Information Systems, Institut Teknologi Del, Laguboti, Sumatera Utara, 22381, Indonesia

Email: mario@del.ac.id

A R T I C L E I N F O

Article history:

Received: Jan 02, 2022
Revised: Jan 14, 2022
Accepted: Feb 08, 2022

Keywords:

Pair Programming, Distributed, Programming Course, Freshmen

A B S T R A C T

Pair programming is an old yet effective method in solving programming problems in the computer science field. Here, at Institut Teknologi Del, we employed this method in the early programming courses to help the students master the learning objectives and practice them by solving problems in a collaborative manner. The COVID-19 pandemic was started in early 2020, it threatened almost every sector in our lives including education. Students were sent home and the learning process was forced to switch from in-person to distancing learning. This situation was a true challenge for our traditional pair programming approach where the students used to work together side-by-side. We then adapt the approach to fit the distributed setting. In this paper, we would like to share our experience in applying distributed pair programming, the problems we faced, and the strategies we employed to achieve the highest learning impacts.

1. Introduction

In early 2020 COVID-19 pandemic stormed every aspect in our lives including the education section. The most recommended strategy to fight the spreading was physical distancing, a disaster to our traditional in-person learning method. Students and lecturers were sent home, and the learning process was forced to switch to the distance learning model. Numerous challenges in the programming context, such as lower productivity, diminished emotional well being [1].

Here at Institut Teknologi Del, the first-year students of computing-related study programs get their first programming introductory courses. In the Information System study program, there are two introductory programming courses namely Visual Programming and Procedural Programming. Visual Programming is available in the first semester and Procedural Programming is in the second semester. Both courses are weighted 2 credits or equal to approximately 6 commitment hours per week. These 6 hours are then divided into 3 activities: a theoretical session (2 hours), a practical session (2 hours), and a tutorial or self-study (2 hours).

In Visual Programming, we introduced the fundamental concepts and constructs of programming through abstractions (visual objects or models). We avoided writing actual code in a particular programming language. The problems and case studies provided in this course were relatively simple. Procedural Programming was more advanced than Visual Programming. In Procedural Programming, more advanced concepts and structures were introduced, the given problems and case studies were much complex, and the use of C language. To achieve the best outcome on these two courses, the students were expected to explore and practice a lot, solving trivial to complex problems, and exchange ideas with other students.

Before the pandemic, the learning process was done in person for both theoretical, practical, and tutorial sessions. The students were grouped in pairs formed at the beginning of the semester. These groups were expected to work as a team to tackle the given problems or case studies [1, 2]. The practical sessions were led by four lab assistants. These lab assistants were senior students who achieved high grades on the subject. The lab sessions were conducted in a pair programming fashion where the students sit side-by-side with his/her pair discussing the given problem, designing the optimal solution, and writing code together on a computer. One student would act as the “Driver” while the other as the “Navigator” role. Studies suggest that pair

Accredited “Rank 4”(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

programming was effective both in the normal in-person setting and remote setting [2], [3]. When applied correctly, pair programming would also be beneficial in both industry [4] and education settings [5]–[9]. Starting from the academic year 2020/2021, applying traditional pair programming was no longer possible since the students were located in dispersed locations, an adaptation to this approach was needed.

A previous study [10] suggested the need for an effective collaborative method in learning programming in a distance context. Moreover, the survey also mentioned the importance of a supportive environment around the students which aligned to [1].

In this paper, we would like to share our 2.5 years of experience in conducting effective pair programming in a distributed setting to achieve the best possible outcome. We believe this paper would give useful insight and answer some challenges mentioned in the previous studies. First, we will describe how we set up the class, how we adapted the pair programming to fit with the distributed setting, the problems we faced, and the strategies we employed to answer the problems.

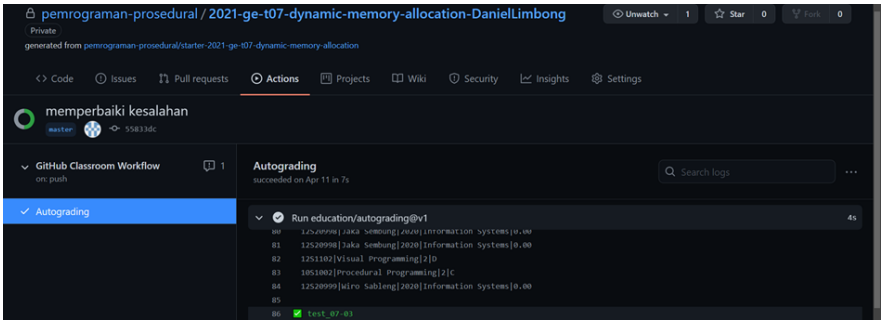

Figure 1. Programming assignments were submitted and graded using GitHub Classroom

2. Method

This research was conducted from the academic year 2019/2020 to the first semester of the academic year 2021/2022. As been stated before, Visual Programming was scheduled for the first semester and Procedural Programming was set for the second semester, hence during the research period, Procedural Programming was not started yet. Table 1 depicts the student distribution during those periods.

TABLE 1
THE STUDENT DISTRIBUTION FROM THE 2019/2020 TO THE FIRST SEMESTER OF 2021/2022 ACADEMIC YEAR

Student Type	VP 2019/2020	PP 2019/2020	VP 2020/2021	PP 2020/2021	VP 2021/2022
Freshmen	62	62	54	53	57
Retake	0	0	0	7	0

Adapting Pair Programming. We adapted the 11 traditional pair programming guidelines suggested in [9], [11]. Some tweaks were introduced to fit the distributed setting but maintain the original goals. For example, the guideline asked the teaching team to closely monitor the application of pair programming and to give immediate feedback which was not feasible due to limited resources. The solution was to ask the students to record their first 30 minutes meeting for a later check and feedbacks.

Group Forming Strategies. To form the group (pairs) we tried two approaches: a purely random approach and clustered based on the students' programming traits (previous programming assessments). The pure random approach was applied to Visual Programming in the academic year 2019/2020 and 2021/2022 whereas the clustered approach was applied to Procedural Programming also in the academic year 2019/2020 and 2021/2022.

Flipping the class. Starting from the academic year 2020/2021, we employed the flipped learning approach. Materials were made available a week beforehand and accompanied by learning videos posted in YouTube [12]. The students were expected to do a self-study before the scheduled weekly meeting. For both Visual Programming and Procedural Programming, the weekly meeting was scheduled for 100 minutes. The first 50 minutes were used to do a small quiz and discuss the previous lab work. The other 50 minutes were used for examining new materials. The lab session was scheduled for 120 minutes where the students were

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

expected to solve a given problem. The students sent their solutions to GitHub Classroom [13], [14] platform for grading (Figure 1). Some tests were prepared and run against the submitted solutions. The grades was calculated instantly based on the test results.

Measurements. In this paper, we used the final grades to measure the effectiveness of pair programming. The final grades of the academic year 2019/2020 were set as the baseline for the pair programming (traditional), the final grades of the academic year 2020/2021 reflected the implementation of the adapted pair programming (distributed), and the final grades of the first semester of 2021/2022 as the improved implementation of the adapted pair programming (also distributed) – Table 2.

TABLE 1
THE STUDENT DISTRIBUTION FROM THE 2019/2020 TO THE FIRST SEMESTER OF 2021/2022 ACADEMIC YEAR

	VP 2019/2020	PP 2019/2020	VP 2020/2021	PP 2020/2021	VP 2021/2022
Grade role	Baseline	Baseline	Initial DPP	Initial DPP	Improved DPP
Forming strategy	Random	Random	Random	Clustered	Clustered

For a better understanding, we also collected students' impressions on the implementation of pair programming, their contribution level, and what problems they faced. This information would surely help us design an even better adaptation in the future.

3. Result and Discussion

First, we would like to compare the final grades between the baseline (traditional pair programming) and the initial adaptation to fit the distributed setting (distributed pair programming). Figure 2 depicts the comparison between the two. In this case, the distributed pair programming seemed did not give a better outcome in the first semester (Visual Programming) but brought a significantly better impact on the second semester (Procedural Programming). This was due to the fact that the Procedural Programming in the academic year 2019/2020 was affected by the distance learning policy.

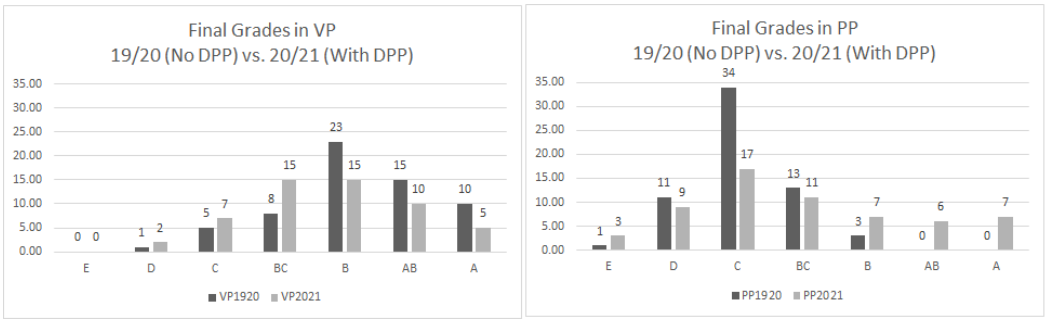

Figure 2. The Impact of the Adopted Pair Programming to Fit the Distributed Setting (DPP) and the Traditional Pair Programming.

From the students' satisfaction perspective, the pure random group forming showed a higher students' satisfaction compared to the clustered approach (**Error! Reference source not found.**). We found that those who were dissatisfied were students within the lower cluster or who had a relatively low programming trait. Students in the upper cluster, on the other hand, were satisfied with the grouping. The clustered approach was first applied in the second semester of the academic year 2020/2021 and based on the positive impact (final grades, **Error! Reference source not found.**) we applied this approach even earlier on the next academic year.

Another strategy was providing immediate feedback as soon as possible. During the lab session, the students were left alone for 60 minutes to try solving the problems by themselves and submit the proposed solution to the GitHub Classroom to get feedback in the form of grades. Afterward, the lab assistants then check the grading before opening a meeting room for groups who failed in most tests. A lab assistant was

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

assigned to handle 5 to 6 groups. Each group would have only 10 minutes to discuss with the assigned lab assistant. These short meetings were aimed to guide them in solving the given problem.

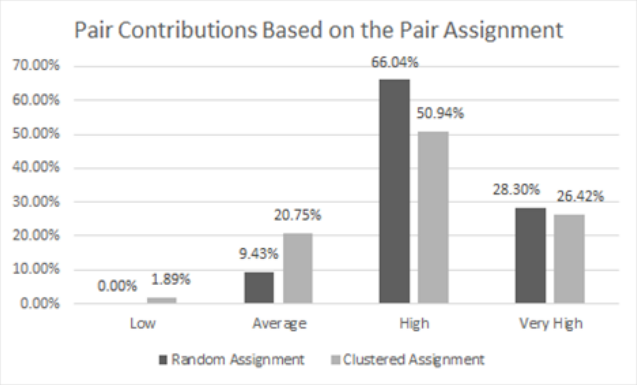

Figure 3. Pair contributions on the programming assignments

From the survey, we found that pair commitment was the biggest issue. This was a serious issue since lack of commitment would threaten the team cohesiveness, motivation, and productivity. To tackle this, we asked the students to interview their pairs as their first task once the groups were formed. We also conducted private meetings with pairs who had issues with personnel commitment to find the resolution. In an extreme situation, we regrouped by isolating the less committing students. By doing so, only students with high commitment work together in a team.

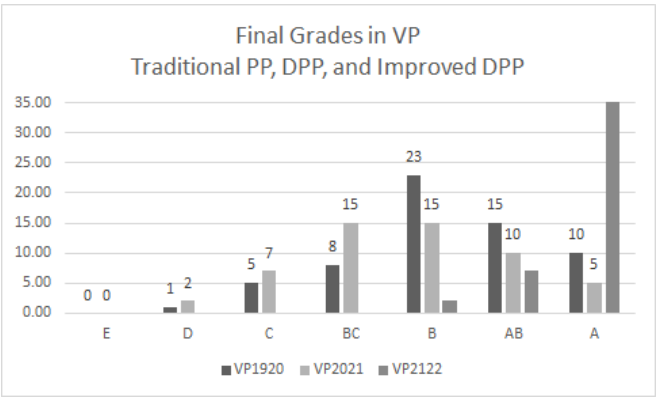

Figure 4. The effectiveness of improved DPP

The combination of three strategies was proved to be effective. The first one is applying the clustered grouping. It helps us to identify which group would likely need extra attention. The second strategy, providing immediate feedback through short meetings did help those who were sitting on the lower cluster. Our last strategy was to resolve the problem of low commitment. The outcome was highly significant, majority of the students reach the highest grade (**Error! Reference source not found.**).

4. Conclusion

Pair programming is an old-style learning method that has been proven to be effective. Unfortunately, this approach was not initially designed for distance learning with pairs located in a distributed setting. It becomes even more challenging to apply this approach to the freshmen students. In this paper, we shared our experience in adapting and applying pair programming to fit the distributed setting.

After 2.5 years of applying pair programming, we come out with three important strategies to achieve the highest student outcome. The first strategy is to form the group using the clustering approach where the students are grouped based on their level of programming traits. This will help us identify and separate low

Accredited "Rank 4"(Sinta 4), DIKTI, No. 36/E/KPT/2019, December 13th 2019.

potential groups from the high potential groups. The low potential groups then will get more attention from the teaching team.

The second strategy is by providing immediate feedback to groups with difficulties. This feedback is an addition to the instant grading from the GitHub Classroom. This strategy is highly beneficial for groups that lied in the lower cluster. Our 10 minutes private meetings worked and did help the students to spot and tackle their mistakes.

The third strategy is by resolving the students' commitment problem. As we have been previously discussed, students with low commitment would eventually affect the group performance, hence serious action must be taken. In our case, when a group member reported this issue, a private meeting was commenced led by one of the teaching team to resolve it. Sometimes this issue was about personnel chemistry and distractions at home. To improve the personnel chemistry, we asked the students to interview their pair. This made the students aware of their pair's personality, characteristics, and traits.

All the above strategies were proven to be effective and did help our students achieve the best outcome (Figure 4). These also tried to answer problems and suggestions mentioned in the previous studies [1], [10]. However, applying all three strategies were uneasy. It demanded time commitment and a significant amount of effort..

5. References

[1] P. Ralph et al., "Pandemic Programming," Empir. Softw. Eng., vol. 25, no. 6, pp. 4927–4961, Nov. 2020, doi: 10.1007/s10664-020-09875-y.
[2] L. Williams and R. R. Kessler, Pair Programming Illuminated. Addison-Wesley Professional, 2003.
[3] K. Beck and C. Andres, Extreme Programming Explained: Embrace Change (2nd Edition). Addison-Wesley Professional, 2004.
[4] M. Rajpal, "Effective distributed Pair Programming," in Proceedings of the 13th International Conference on Global Software Engineering, New York, NY, USA, May 2018, pp. 6–10. doi: 10.1145/3196369.3196388.
[5] C. McDowell, L. Werner, H. Bullock, and J. Fernald, "The Effects of Pair-Programming on Performance in An Introductory Programming Course," in Proceedings of the 33rd SIGCSE technical symposium on Computer science education, Cincinnati, Kentucky, Feb. 2002, pp. 38–42. doi: 10.1145/563340.563353.
[6] T. Benadé and J. Liebenberg, "Pair Programming as a Learning Method Beyond the Context of Programming," in Proceedings of the 6th Computer Science Education Research Conference, New York, NY, USA, Nov. 2017, pp. 48–55. doi: 10.1145/3162087.3162098.
[7] L. Williams and R. L. Upchurch, "In support of student pair-programming," ACM SIGCSE Bull., vol. 33, no. 1, pp. 327–331, Feb. 2001, doi: 10.1145/366413.364614.
[8] Y. Lu, X. Mao, T. Wang, G. Yin, and Z. Li, "Improving students' programming quality with the continuous inspection process: a social coding perspective," Front. Comput. Sci., vol. 14, no. 5, p. 145205, Jan. 2020, doi: 10.1007/s11704-019-9023-2.
[9] L. Williams, D. S. McCrickard, L. Layman, and K. Hussein, "Eleven guidelines for implementing pair programming in the classroom," in Agile 2008 Conference, 2008, pp. 445–452.
[10] C. Lacave and A. I. Molina, "The Impact of COVID-19 in Collaborative Programming. Understanding the Needs of Undergraduate Computer Science Students," Electronics, vol. 10, no. 14, Art. no. 14, Jan. 2021, doi: 10.3390/electronics10141728.
[11] J. Bevan, L. Werner, and C. McDowell, "Guidelines for The Use of Pair Programming in A Freshman Programming Class," in Proceedings 15th Conference on Software Engineering Education and Training (CSEE T 2002), Feb. 2002, pp. 100–107. doi: 10.1109/CSEE.2002.995202.
[12] "YouTube." https://www.youtube.com/ (accessed Dec. 23, 2021).
[13] C.-Z. Kertész, "Using GitHub in the Classroom – a Collaborative Learning Experience," p. 6, 2015.
[14] "GitHub Classroom." https://classroom.github.com/ (accessed Dec. 24, 2021).