2ZH-07

Merging Mob Programming and a Student Mentor Method into a Programming Class

Shota Kaieda　Daisuke Saito　Hironori Washizaki　Yoshiaki Fukazawa

Waseda University

ABSTRACT

Learning to program is perceived as difficult. Different approaches have been implemented in classes to solve this issue. Here, we propose a collaborative learning method that applies the mob programming technique and utilizes a student mentoring method. A workshop for middle school students in which they completed a project revealed positive impressions of this method. Student feedback recognized that the project was completed due to the help of the student mentor and communication with peers. Our method allows cooperative behavior between experienced and novice students. We plan to continue with this approach and to investigate the learning effectiveness of our method.

1　INTRODUCTION AND PROBLEM

Learning to program is difficult for students since many skills must be learned [1]. Often, people tend to collaborate with their colleagues when a difficult problem is encountered. Thus, we introduce a collaborating learning method that uses mob programming (MP), which incorporates a student-centric mentoring method. In collaborative learning, students with different knowledge levels work together toward a common objective [2]. MP is a collaborative working technique where a team of more than two people rotates coding duties on one computer at the same place and time [3]. This method is designed to help novice programmers learn basic programming concepts through collaboration, while simultaneously improving the mentor's teaching and communication skills.

MP alone involving only novice programmers without prior training and knowledge will not result in a gain of skill. Consequently, the mentor method is proposed. In the MP study group, students with programming experience become the mentor, and the remaining students are the mentees. The mentor's task is to teach the mentees programming. The mentees' tasks are to propose new ideas and create discussion points during the MP session.

We propose two research questions (RQs). RQ1: Can a student mentor method be used in MP to complete a project? RQ2: What are the students' impressions of MP? RQ1 demonstrates the viability of the proposed method. RQ2 determines the pros and cons of the proposed method. These two RQs are answered through project creation by the students and a survey.

2　BACKGROUND AND RELATED WORK

MP is mainly used for software development. However, there are cases where it has been used to support the learning of technical skills. In one case, MP was used to train a group of novice developers, improving not only their coding skills but also other aspects such as their ability to focus and deployment procedures [4]. In another case, MP was integrated into the educational field [5]. In the integrated set up, the students were collaborating online. We suspect that taking MP online sacrifices the strength of collaboration and discussions that occur face-to-face.

3　METHOD

A MP group is created with three to four students, where at least one student in each group has programming experience. The student with programming experience is assigned as the mentor. The mentor provides support to the mentees through hints on implementations, live code explanations, and other means necessary, which are at the mentor's discretion. The mentees actively engage in the discussion to gain new knowledge. The role of the mentor is assigned to the student with programming experience, which is denoted as S1 in Fig. 1. It is not required to have the mentor as the first coder.

Figure 1 overviews the setup. A display is setup for each MP group so the students can follow the coder. The students rotate every five minutes. The coder is the student directly in front of the PC. Only the student in the coder role can code. The other students must wait their turn to code.

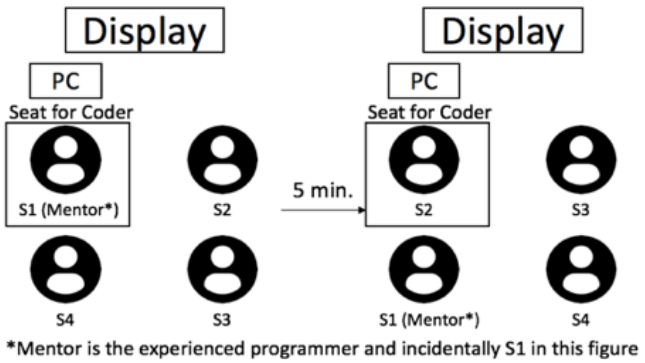

Figure 1 Mob programming setup

Merging Mob Programming and a Student Mentor Method into a Programming Class
Shota Kaieda, Waseda University
Daisuke Saito, Waseda University

4 EXPERIMENT

An eighty-minute workshop was conducted with nineteen middle school students. Six students had prior experience in programming. Their programming experience ranged from a month to 3.5 years. The workshop was allocated as follows: 10 minutes to become accustomed to Scratch, 60 minutes for MP, and 10 minutes to answer a survey. The objective of each group was to create a game using Scratch that met the given requirements. The requirement objectives were to expose novice programmers to several different programming concepts such as the use of variables, conditionals, controls, etc. The requirements were as follows: (1) The game must contain an in-game timer. (2) The game must contain some sort of point system. (3) At the end of the game, some sort of action must be executed. The mentor presented the programming concepts to novice programmers by following our MP method, which is described above. At the end, the students' impressions of MP were collected through a survey with a free response section. Some of the questions inquired about whether MP learning was fun, rating the impression of MP, and a free-response on how they felt towards MP.

5 RESULTS

Of the five groups, three were able to create a game that met most or all of the requirements. The three successful groups displayed similar traits. The mentor and the group were communicating a lot, and the mentor even provided instructions on how to proceed when not the coder. Figure 2 shows screenshots of the games created by students.

Figure 2 Games created by students

The left is a baseball game where a ball is thrown, and if the batter hits the ball, a score is rewarded. The right is a basket game where a point is rewarded if the orange is caught and a point is deducted if hit by the lightning bolt. All the students provided positive survey results regarding their enjoyment of MP. 79% of the impressions of MP were positive. This high impression, which was deduced from the free response answers in the survey (Table 1), was attributed to the high communication and cooperation between the students.

The workflow efficiency depended on the mentor's guidance skills. For the mentees, if all went well, communication and idea-sharing occurred, but some students still found programming to be difficult. The completed and semi-completed games created by the students confirmed that projects can be finished via a student mentoring method and MP. This finding affirms RQ1: Can a student mentor method be used in MP to complete a project? The survey also indicated that the overall positive impression towards MP is due to the communication and teamwork among the group members. This answers RQ2: What are the students' impressions of MP?

Table 1. Survey Free Responses

Mentors	"Listening to every member's opinion saved time." "Work went smoothly when cooperating after a discussion." "It was a great opportunity to have communication with everyone in the group" "Because I couldn't guide them well, work did not go smoothly."
Mentees	"Because I was able to communicate with others, we were able to teach skills to those who did not understand." "Cooperating with the team made it easy." "Because the mentor gave specific instructions, work went smoothly, and we were able to create a high-level game." "The experience was fun; however, I had a hard time understanding since it was my first time."

6 CONTRIBUTIONS AND FUTURE WORK

We demonstrated that the student mentor method can effectively introduce beginner programmers to programming. The positive impression of MP is due to the cooperation and communication among students. In future research, we will investigate the effectiveness and the detailed role of the student mentor, including examining how the student mentor interacts with members in the MP group and how the teaching style of the student mentor affects the results of the group.

REFERENCES

[1] Essi Lahtinen, Kirsti Ala-Mutka, and Hannu-Matti Järvinen. 2005. A study of the difficulties of novice programmers. In Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education (ITiCSE '05). ACM, New York, NY, 14-18. DOI: https://doi.org/10.1145/1067445.1067453

[2] Gokhale A.A. (2012) Collaborative Learning and Critical Thinking. In: Seel N.M. (eds) Encyclopedia of the Sciences of Learning. Springer, Boston, MA

[3] M. Shiraishi, H. Washizaki, Y. Fukazawa and J. Yoder, "Mob Programming: A Systematic Literature Review," 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), Milwaukee, WI, USA, 2019, pp. 616-621. https://doi: 10.1109/COMPSAC.2019.10276

[4] Boekhout K. (2016) Mob Programming: Find Fun Faster. In: Sharp H., Hall T. (eds) Agile Processes, in Software Engineering, and Extreme Programming. XP 2016. Lecture Notes in Business Information Processing, vol 251. Springer, Cham

[5] Sreecharan Sankaranarayanan, Xu Wang, Cameron Dashti, Marshall An, Clarence Ngoh, Michael Hilton, Majd Sakr, and Carolyn P. Rosé. 2019. Online Mob Programming: Bridign the 21st Century Workplace and the Classroom. In Proceeding of the 13th annual CSCL conference on computer supported collaborative learning (CSCL '19). Vol. 2 855-856 Lyon, France: International Society of the Learning Sciences.