Jurnal Teknologi Maklumat & Multimedia **8**(2010): 67 - 74

Experimenting with Pair Programming in an Introductory Programming Course

DALIALAH ABD GANI, NORASIAH MOHAMMAD
& ZAINURA IDRUS

ABSTRACT

This paper describes how pair programming is conducted in the collaborative learning of an introductory programming course at University Technology of MARA (UiTM), aimed at increasing students' motivation and confidence. The paper begins by defining the term pair programming and discussing the rationale for bringing industry's pair programming into the classroom, as supported strongly by findings from numerous research. The mechanism for implementation and certain issues that might compromise the effectiveness of the pairing are also discussed. Our findings show that pairing not only does not compromise students' learning but that it enhances students' enjoyment, confidence and quality of programs. Thus, suggesting that pair programming is an effective pedagogical tool for teaching any programming course.

Keywords: Pair Programming, Solo Programming, Pedagogical Tool.

ABSTRAK

Kertas kerja ini membincangkan bagaimana pengaturcaraan berpasangan dilaksanakan ke atas mata pelajaran Pengenalan Kepada Pengaturcaraan di Universiti Teknologi MARA (UiTM) bertujuan meningkatkan motivasi dan keyakinan pelajar. Kertas kerja ini bermula dengan mendefinisikan pengaturcaraan berpasangan. Kemudian disusuli dengan hujah-hujah mengapa perlu menerapkan pengaturcaraan berpasangan yang telah dipraktikkan di industri ke dalam bilik darjah sepertimana yang ditekankan oleh penyelidik-penyelidik terdahulu. Prosedur perlaksanaan dan isu-isu yang timbul dan menggugat keberkesanan pengaturcaraan berpasangan juga dibincangkan. Hasil penyelidikan kami, menunjukkan pengaturcaraan berpasangan bukan sahaja meningkatkan keberkesanan pembelajaran, malah menambah keseronokan, keyakinan dan kualiti perisian. Kesimpulannya, pengaturcaraan berpasangan merupakan cara yang berkesan untuk diguna pakai di dalam bilik darjah untuk sebarang mata pelajaran pengaturcaraan.

Kata kunci: Pengaturcaraan Berpasangan, Pengaturcaraan Solo, Alat Pedagogi.

INTRODUCTION

Pair programming is a style of programming in which two programmers work side-by-side continuously at one computer, on the same design, algorithm, code or test (Wiebe et al. 2003). Pair programming differs from normal two-person team projects, called side-by-side programming (SbS), in its level of collaboration. In SbS a task is assigned to a pair of programmers, and each programmer has his own computer (Nawrocki et al. 2005). This allows them to split the task into subtasks and work on each subtask individually. Team projects are usually divided into 'my' part and 'your' part. However, with collaborative programming, the

entire project is 'ours' (Williams et al. 2001). In pair programming, a task is assigned to a pair of programmers who are equipped with one computer. While one programmer is writing a piece of code, the other is watching, asking some questions, and proposing test cases (Nawrocki et al. 2005).

Within the pair, work is split into two roles, known as the driver and the navigator. The driver is the person at the keyboard, responsible for the actual typing of code. The navigator is an active observer and monitor of the code being written. They are in constant communication, asking and answering questions between one another, brainstorming and negotiating their course of action. They may switch roles frequently in the course of a programming session (Chong et al. 2005).

Although collaboration has been employed in non trivial software development tasks, computer programming has traditionally been taught and practised as a solitary activity (Cockburn et al. 2001). This solitary programming approach has begun to change in recent years. Advocates of collaborative programming have emerged and computer science instructors are increasingly experimenting, either formally or informally, with pair programming in the classroom. Recent findings now question the long followed practice of requiring students to complete programming projects individually (Wiebe et al. 2003, McDowell et al. 2003, Williams et al. 2001). In fact, findings from previous studies indicate that pair programming has a positive impact on students' performance, confidence and enjoyment while none demonstrated that student learning is compromised (McDowell et al. 2003).

This paper is organized as such Section 2 covers benefits of pair programming, Section 3 discusses methods of data collection, Section 4 analyses data to produce result, Section 5 is the conclusion and lastly Section 6 is the acknowledgements.

BENEFITS OF PAIR PROGRAMMING

Previous studies of pair programming in university programming classes have shown that pair programming yields better design, more compact code and fewer defects, offer greater confidence and more enjoyment of the programming process (Chong et al. 2005, McDowell et al. 2003, Williams et al. 2001).

In another study conducted by Cockburn and Williams, the general benefits of pair programming in the classroom are cited as follows (Cockburn et al. 2001):

i. Increased discipline
   Pairing partners are more likely to "do the right thing" and are less likely to take long breaks. Two people working together in a pair treat their shared time as more valuable. They tend to cut phone calls short; they don't check e-mail messages or favorite Web pages; they don't waste each others time.

ii. Better code
   One of the programmers, the driver, has control of the keyboard/mouse and actively implements the program. The other programmer, the observer, continuously observes the work of the driver to identify tactical (syntactic, spelling, etc.) defects, and also thinks strategically about the direction of the work. On demand, the two programmers can brainstorm any challenging problem.

iii. Improved morale
   Pair programming, done well is much more enjoyable than programming alone, done well.

iv. Collective code ownership
   When everyone on a project is doing pair programming, and pairs rotate frequently, everybody gains a working knowledge of the entire codebase. Since the two programmers periodically switch roles, they work together as equals to develop software.

v. Mentoring
Everyone including junior programmers, has knowledge that others don't. Pair programming is a painless way of spreading that knowledge. This is a very effective way for a more experienced person to teach a less experienced person.

vi. Team cohesion
People get to know each other more quickly through pair programming and it encourages team jelling.

vii. Fewer interruptions
People are more reluctant to interrupt a pair than they are to interrupt someone working alone.

Pair programming conducted in the proper manner is shown beneficial to both the learner and the teacher. It enhances student learning but with less assignments to be graded by the instructors.

METHODOLOGY

The purpose of this study was to implement pair programming in an introductory programming class and investigate the impact on students' confidence, satisfaction, enjoyment and program quality when working in a pair. This study is part of a larger study to assess the effectiveness of pair programming on students' performance and problem solving skill for the whole academic year from July 2008 to Jun 2009. The results reported in this paper are based on a subsection of the whole study.

Sample for this study were first year students enrolled in Algorithm Fundamentals (ITC420) intended primarily for those pursuing information technology related majors. From 106 students who registered for ITC420 during semester July – October 2008, 71 students were used in this study which is approximately 67% of the total enrollment.

Pair programming was used for 51 students in two separate classes. The first class had 28 students, all work in pairs but, the second class had 23 students, thus 20 of them work in pairs but the remaining 3 students were teamed to work collaboratively together. The solo programming class of 20 students was used as the control group so that comparisons can be made between the performance of the students' for pair programming and solo programming.

On the first day of class students were given a briefing on pair programming, how it would be implemented and the rules they had to observe. The difference between pair programming and team project was highlighted to avoid the misunderstanding that they can partition the problems and work on these partitions separately. Students in the pairing class submitted a list of three names of potential partners and the pairing was done based on these preferences. Those who did not state any preferences were assigned a partner at random.

Students from both the pair programming and solo programming classes were required to submit four graded programming assignments and were encouraged to complete four additional no graded programming assignments. Together with each assignment, students submitted a log recording their level of confidence, satisfaction and how much they enjoyed doing the assignment. In the pair programming classes, students were required to complete all assignments using pair programming while in the solo programming class students were required to complete assignments independently.

The first three weeks of the semester, pairing adjustments were allowed considering intra-pair stress due to incompatibility of characters or experience levels. Significant difference in experience levels would make the pairing less functional. The more experienced student would not be willing to wait for the other 'slow' partner to understand the essential programming concepts and consider pair programming as a waste of time. For serious cases, students can opt out of the pairing section.

Although students learn through two different approaches; solo and pair programming, all sat for their quizzes and tests independently. These independent assessments measured each individual student's knowledge of programming concepts and ability to solve new problems and write new code, thus enabling the researchers to understand the effects of pairing on individual student learning.

RESULTS

The findings in this paper are part of a larger ongoing study to assess the effectiveness of pair programming and in this section we present the results from a subsection of the study. Our focus was investigating the effects of pair programming on students' performance, confidence, satisfaction and enjoyment.

STUDENT PERFORMANCE

This study measured two related, but distinct indicators of course mastery. The first, the quality of the programs that students produce, represented by the students' average score on the graded programming assignments. The second is the extent to which students were able to apply the programming concepts. The quiz and test scores were used to assess students' mastery of the programming concepts when working individually regardless of whether they paired or not.

i. Program Scores

As argued by previous research, the very process of working collaboratively enhances the quality of programs that pairs produce (McDowell et al. 2002). In this paper we compared the average program scores of students in the pairing classes with nonpairing class.

TABLE 1. Program scores

	Mean	Median	Std. Dev.
Pair	80.05	78.30	8.69
Solo	55.91	57.50	23.85

Students' in the pair programming classes reported significantly higher program scores (80%) than students in the solo programming class (55.9%). Looking at the standard deviation of both, it can be seen that the program scores for solo programming are much more widely dispersed from the mean (23.9) compared to pair programming (8.69).

Figure 1 shows the average program scores for students of pair programming while the graph below, Figure 2 shows the average program scores for students of solo programming. From the two graphs it can be seen that the program scores for solo programming is more widely spread compared to the collaborative effect of pair programming that narrow the gap between the weak and the strong students.

ii. Quizzes and Test Scores

The reluctance to use pair programming in the classroom may be due to a concern that at least some students will earn grades that actually reflect their partner's work. It is possible that the pair programming students in this study obtained higher average programming scores because weaker students receive scores due to the work of the stronger student in the pair and cause the average programming scores to increase artificially.

Thus, it is important to analyse not only students' performance when working in pairs but also when they work independently. The following analysis will help us to understand the effects of pair

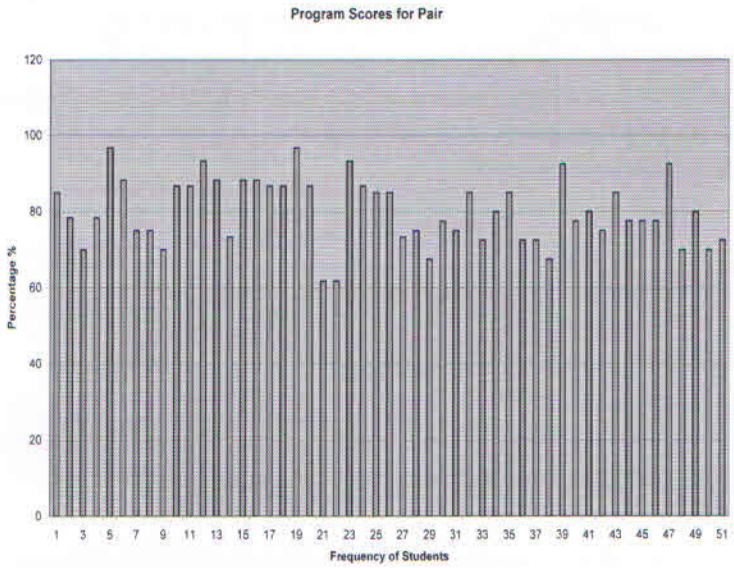

FIGURE 1. Program scores for pair

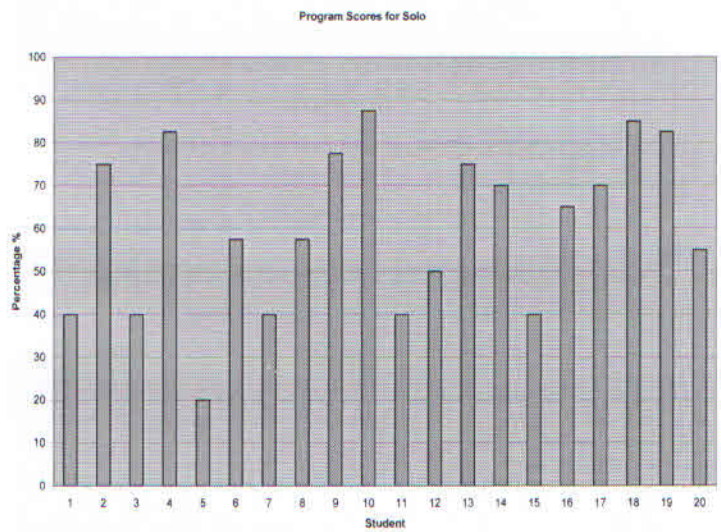

FIGURE 2. Program scores for solo

programming on individual student learning. Since all participants in this study took the quizzes and test independently, the average quizzes and test scores were used as indicators as to how much they mastered the subject matter.

TABLE 2. Quizzes and test scores

	Mean	Median	Std. Dev.
Pair	72.73	71.90	15.13
Solo	65.50	64.00	15.84

The pair programming recorded higher average (72.7%) on the quizzes and test scores compared with solo programming (66.5%). The standard deviations were about the same except a bit higher for solo programming Overall, pair programming students performed better than the solo programming students. This

finding suggests that a student's ability to independently apply programming concepts not compromised by learning to program in pairs.

Figure 3 shows the average quizzes and test scores for students of pair programming while Figure 4 shows the average quizzes and test scores for students of solo programming. More paired students achieve 80% and above as compared to solo students.

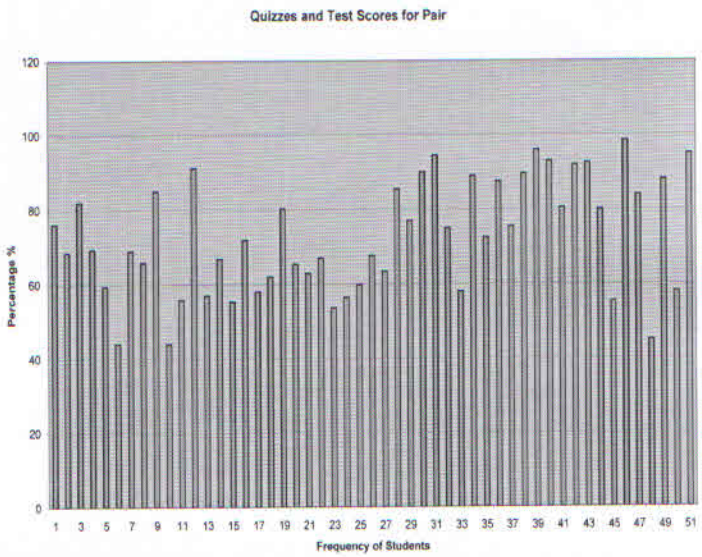

FIGURE 3. Quizzes and test scores for pair

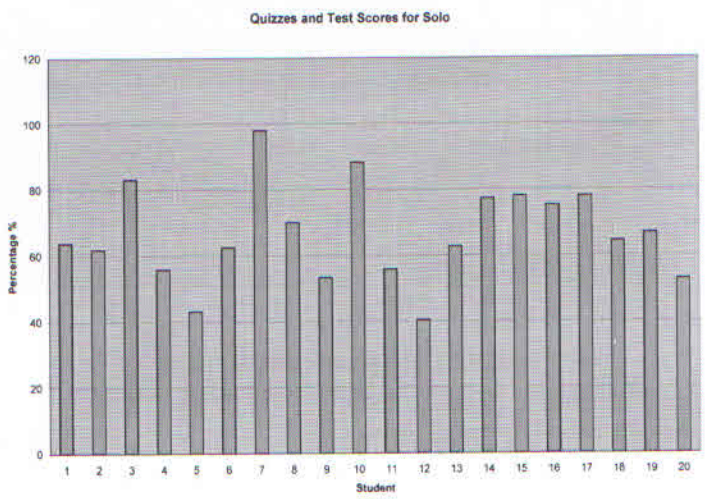

FIGURE 4. Quizzes and test scores for solo

CONFIDENCE, SATISFACTION AND ENJOYMENT

The most important goal for students in any class is mastery of the subject matter. This is especially true for introductory courses where success in further advanced courses is dependent on a strong foundational knowledge.

However, subjective experiences in introductory courses may contribute to students' motivation and interest in understanding the basic programming concepts and encourage them to persistently work on the programming solutions. Thus, it is important to understand how pairing experience influences students' confidence, enjoyment and satisfaction with their programming solutions.

Each submission of a program had to include students' answers to the questions as listed in Table 3. Students' response to the questions were based on the Likert scale from 0 (least confident/enjoyable/satisfied) to 10 (very confident/enjoyable/satisfied).

TABLE 3. Questions asked about each program

Confidence	How confident are you in your solution?
Satisfaction	How satisfied are you with the time and effort you (solo or paired) spent on the assignment?
Enjoyment	How much did you enjoy working on this assignment?

Responses to the above specific questions are presented by separate tables in the following sub sections.

i. Students Confidence

Overall, students in the pair programming classes reported significantly higher confidence in their programming solutions (83.5%) than students in the nonpairing or solo programming classes (66%).

TABLE 4. Students' confidence

	Mean	Median	Std. Dev.
Pair	85.53	85.00	12.66
Solo	66.00	69.00	18.16

ii. Students Satisfaction

Similarly, students in the pair programming classes reported much higher satisfaction (83.9%) with the effort and time spent on the programming assignments than students in the solo programming class (68.7%).

TABLE 5. Students satisfaction

	Mean	Median	Std. Dev.
Pair	83.92	85.00	11.12
Solo	68.68	70.00	18.12

iii. Students Enjoyment

Enjoyment level of the students working on the programming assignments were also reported higher for the pair programming classes (82.3%) compared to the solo programming classes (66.6%).

TABLE 6. Students' enjoyment

	Mean	Median	Std. Dev.
Pair	82.25	85.00	13.82
Solo	66.59	70.00	17.62

CONCLUSION

The results of this study provide some convincing evidence of the effectiveness of pair programming as a pedagogical tool. Students, who paired produced better programs, were more confident in their work, more satisfied with their solutions and enjoyed learning programming more. We hope these findings will encourage lecturers to use pair programming not only in introductory programming courses but also in more advanced programming courses.

ACKNOWLEDGEMENTS

We wish to thank Chairman of the school of Computer Science at UiTM Shah Alam, Associate Professor Dr Naimah Mohd Hussin for her constructive criticisms and suggestions before the full paper was completed. Our gratitude is also extended to our collegue Associate Professor Norshahida Shaadan for reading through the extended abstract and commenting especially on the analysis of data. Last but not least praises to Allah for making this fruitful study possible.

REFERENCES

Chong, J., Plummer, R., Leifer, L., Klemmer, S., Eris, O. & Toye, G. 2005. Pair, Programming: When and Why it Works, *17th Workshop of the Psychology of Programming Interest Group*, Sussex University, 2005.

Cockburn, A. & Williams, L. 2001. *The Costs and Benefits of Pair Programming, Extreme Programming Examined.* New York: Addison Wesley-Longman, 2001.

McDowell, C., Werner, L., Bullock, H. & Fernald, J. 2002. The Effects of Pair Programming in an Introductory Programming Course, *in 33rd SIGCSE Technical Symposium on Computer Science Education*, 2002. Northern Kentucky: ACM Press.

McDowell, C., Werner, L., Bullock, H. & Fernald, J. 2003. The Effects of Pair Programming on Student Performance and Pursuit of Computer Science Related Majors, *International Conference on Software Engineering*, Portland, Oregon, USA, 2003.

Nawrocki, J., Jasinski, M., Olek, L. & Lange, B. 2005. Pair Programming vs. Side-by-Side Programming, Proceedings of EuroSPI, Budapest, 2005.

Wiebe, E., Williams, L., Petlick, J., Nagappan, N., Balik, S., Miller, C. & Ferzli, M. 2003. Pair Programming in Introductory Programming Labs, *Proceedings of the 2003 American Society for Engineering Education Annual Conference & Exposition*, 2003.

Williams, L. A. & Kessler, R. R. 2001. Experiments with Industry's "Pair Programming" Model in the Computer Science Classroom, *Computer Science Education* 11(1): 1-15.

Dalialah Abd. Gani, Norasiah Mohammad & Zainura Idrus
Fakulti Sains Komputer dan Matematik
Universiti Teknologi MARA
40200 Shah Alam, Selangor
dalialah@tmsk.uitm.edu.my
norasiah@tmsk.uitm.edu.my
zainura@tmsk.uitm.edu.my