Increased Retention of Early Computer Science and Software Engineering Students using Pair Programming

Jeffrey C. Carver, Lisa Henderson, Lulu He, Julia Hodges, Donna Reese
Department of Computer Science and Engineering
Mississippi State University
{carver, lisah, lh221, hodges, dreese}@cse.msstate.edu

Abstract

An important problem faced by many Computer Science and Software Engineering programs is declining enrollment. In an effort to reverse that trend at Mississippi State University, we have instituted pair programming for the laboratory exercises in the introductory programming course. This paper describes a study performed to analyze whether using pair programming would increase retention. An important goal of this study was not only to measure increased retention, but to provide insight into why retention increased or decreased. The results of the study showed that retention significantly increased for those students already majoring in Computer Science, Software Engineering, or Computer Engineering. In addition, survey results indicated that the students viewed many aspects of pair programming to be very beneficial to their learning experience.

1. Introduction

Declining enrollment is a common problem faced by Computer Science departments over the past few years. There are many valid explanations for the decline, but the trend is still problematic. In the Computer Science and Engineering Department (CSE) at Mississippi State University (MSU), we have also encountered this trend. Not only has there been an overall decline in the number of students choosing Computer Science-related (CS-related) majors (i.e. Computer Science, Software Engineering and Computer Engineering), but also a decline in the number of female students pursuing degrees in those majors. While the overall trend is not good, the decline in female enrollment is particularly troublesome. Because of the declining enrollment, it has become even more important to devote effort to retaining as many students who have chosen a CS-related major as possible.

Previous work (discussed in Section 2) indicates that the use of pair programming in introductory computer science courses may help reverse this trend. Building on these previous findings, we instituted pair programming for the laboratory exercises in the introductory programming courses beginning in the fall 2005 semester. Section 3 provides a more detailed description of the course, the use of pair programming in that course, and the study design to evaluate its effect. The top-level results discussed in Section 4 show that the retention rate of students enrolled in the course during the time pair programming was used was higher than in previous years. To more fully understand whether the use of pair programming was a major contributor to that increased retention, the students in the pair programming course were asked a series of questions about their experiences. The responses to these questions are also analyzed in Section 4. Section 5 summarizes the findings and discusses the conclusions that were drawn from the study.

2. Background and Related Work

Computer Science faculty members are increasingly experimenting, either formally or informally, with pair programming in the classroom. Most of the research has focused on establishing the efficacy of the practice for educating students. Accumulated evidence indicates that pair programming leads to higher student achievement, a more positive attitude toward the subject, improved confidence and enjoyment in the class, and a variety of other benefits [3, 5, 6, 8, 9].

In addition to studying the effects of pair programming on student performance and the students' subjective experiences with pairing, researchers are also interested in the effect that pairing might have on the retention of students in CS-related majors. Specifically, they wanted to know if using pair programming as a learning tool for beginning programmers influences the students' decisions to choose a CS-related major. Results indicated that pair programming students were more likely to continue with further computer science classes and to declare a CS-related major than students who did not pair program [10, 11].

Interestingly, although pair programming was beneficial for all types of students, it appeared to be particularly true for women. Confidence and retention in CS-related majors increased more for female students who paired than it did for men [2]. Significantly, more of the female students who worked in pairs went on to declare a CS-related major than female students who programmed solo [4]. One possible explanation for this increase is that pair programming changes the atmosphere of the classroom, making it more supportive and less competitive. It helps to change the long-held belief that work in the information technology field is conducted in a competitive rather than collaborative environment. It is believed that competition rather than collaboration is one of the reasons why fewer women are majoring in computer science [7].

3. The Study

Section 3.1 provides a brief overview of the course in which the study was conducted. Section 3.2 then describes the study designed to understand the impact of pair programming.

3.1 Setting

This study was conducted in the Introduction to Computer Programming course at MSU. This course is the first course in the sequence of courses required for all Computer Science, Software Engineering and Computer Engineering (CS/SE/CE) students. The course is generally taken during a student's first semester. The main objectives of the course are:

1. To introduce principles and practice of software development using the object-oriented programming approach.
2. To develop the problem solving skills necessary to develop software solutions to problems.
3. To develop knowledge of the data and control structures available in the object-oriented programming paradigm and their appropriate uses.[1]

The course is taught using the C++ language. In addition to the 3 hours of lecture each week, the students are required to participate in a 3-hour laboratory session. During the laboratory session, the students write programs to help reinforce the material covered during lecture.

In the fall 2005 semester, we made a change to the structure of the course. In previous semesters, the students did the laboratory programming assignments individually. In fact, discussion of the assignment or solution with other students in the course was considered cheating. They were allowed to receive help only from the graduate teaching assistant or from the instructor. The change introduced in the fall 2005 semester was to require the students to

[1] Taken from the ABET syllabus for the course

use pair programming during the laboratory assignments. Each student was assigned a partner and was required to perform the laboratory assignments with that partner. The practice of pair programming is one of the important aspects of eXtreme Programming [1]. In pair programming, two people work together on the same computer to develop the software. The programmers take turns as the "driver" (i.e. the person who is typing on the keyboard) and the "navigator" (i.e. the person who is reviewing what the driver types in real-time and providing input to correct problems).

The partner assignment was done as follows. At the beginning of the course, we had no information about the students' technical ability, but we did have their Myers-Briggs personality type. So, the first set of pairing was done pseudo-randomly to ensure that some pairings had the same Myers-Briggs personality type and some had different personality types. When this data was analyzed after the study, personality type had no effect on the results, so these pairings can be treated as random. After completion of the fourth laboratory exercise, the partners were reassigned. By this point in the semester, the students had taken some exams. So, following advice from previous research that students work better with a partner of equal ability, we split the class into thirds based on the exam grades. The pairings were then randomly assigned ensuring that the students were both in the same third of the class. Again, the students completed four laboratory exercises with this partner. Finally, the pairings were reassigned a third time (in the same manner as the second pairing) to complete the final four laboratory assignments.

3.2 Study Design

The study had two main goals. The first goal was to determine whether allowing students to work with a partner (pair programming) while doing their laboratory programming assignments would increase the retention rate of students enrolled in a CS-related major at MSU. The secondary goal was to better understand any factors that could impact or explain the results relative to the first goal. The data for this study was collected from students enrolled in different semesters of the Introduction to Computer Programming course. In some of the semesters, the students worked on their laboratory programming assignments in pairs; and in other semesters they worked alone.

To address the first goal (retention), the subject population was examined in two ways. First, we were interested in the percentage of CS/SE/CE majors at the time they took the course who were still CS/SE/CE majors one year later. Second, we were interested in the percentage of all students who took the course who were CS/SE/CE majors one year later. Previous researchers performed similar retention analysis one year after enrolling in a pair programming course [5, 10, 11]. In addition, historical data from MSU indicated that students who were going to change majors typically did so within their first year of study. Therefore, to track retention rates, the variable measured was the percentage of students who were CS/SE/CE majors one year after completion of the introductory programming course. Finally, to be consistent and focus on the proper target population, only the data from students who were enrolled in the course as freshmen was included in the analysis for this goal. For this portion of the study, data was collected from 4 semesters (3 without pair programming and 1 with pair programming) as shown in Table 1.

To address the second goal and provide insight into the results for Goal 1, the students in the fall 2005 semester were given a series of surveys throughout the semester. These surveys were anonymous; therefore this data was not able to be limited to only freshmen or only subjects with particular majors as was done for Goal 1. Two different types of surveys were used. The goal of these surveys was to determine whether the students found pair programming useful and beneficial and whether they believed it helped them in the course.

Table 1 – Study Design

Semester	CS/SE/CE Students	All Students	Used Pair Programming?
Spring 2004	3	7	NO
Fall 2004	48	52	NO
Spring 2005	5	20	NO
Fall 2005	48	75	YES

The first set of surveys, the *Partner Evaluation Surveys*, allowed the students to provide confidential feedback about their partner. This survey was given two times (to evaluate two different partners). Due to course logistics, this survey was not given after the third pairing. For each statement on the survey, the students indicated their level of agreement using the following five-point scale: 1 – Strongly Disagree; 2 – Disagree; 3 – Neither Agree Nor Disagree; 4 – Agree; 5 – Strongly Agree. The statements included on this survey were:

1. My partner and I got along well while working on programming assignments.
2. I learned about concepts covered in the lecture portion of the course by working with this partner on the programming assignments.
3. I gained more understanding of concepts in the course by explaining them to this partner.
4. I was more efficient in debugging my code while working with this partner on the programming assignments.

The second set of surveys, the *Post-lab Surveys*, allowed the students to comment on how well they worked with their partner during a specific programming assignment. Again, due to course logistics, this survey was completed after only three laboratory assignments throughout the semester (after lab assignments 1, 2 and 6). The questions (using the same five-point scale as used in the first survey) were:

1. The work of writing the program was split evenly between my partner and me.
2. My partner and I took equal turns as driver and navigator.
3. I learned more working with a partner on this programming assignment than I would have without a partner.

4. Results and Discussion

This section first provides the results for the top-level goal of improved retention. Next, the data collected on the surveys throughout the pair programming semester is analyzed to provide additional insight into the main result.

4.1 Retention

The primary goal of this study was to understand the impact of pair programming on the retention of students. As described in Section 3, the analysis was done by examining the percentage of students majoring in CS/SE/CE one year after completion of the course. Therefore, to understand whether pair programming improved retention, data from four semesters was used: The spring 2004, fall 2004, spring 2005 semesters (no pair programming) and the fall 2005 (pair programming) as illustrated in Table 1.

The data (presented in Table 2) was analyzed in different ways. First, to address the main goal of the study, only the freshman students who were already CS/SE/CE majors were included. This analysis was done to determine whether using pair programming would encourage those students who had already declared a CS/SE/CE major not to change to another major. Then, to determine the impact on two important sub-groups, the analysis was re-done for only the female students and again for only minority students. In each case, a

Pearson's chi-square test was conducted to determine if the proportion of students who were CS/SE/CE majors 1 year later was significantly higher for the students who had used pair programming than for the students who had not used pair programming. There is an increase in the percentage of students who remain CS/SE/CE majors for all three cases (all students, female only, minority only), with the overall difference being significant (p = .014). For female students, the difference in the retention percentages is large (66.7% to 33.3%), but there are too few data points for this result to be significant. This trend is promising and will be monitored as more data is collected.

In addition, the data for all freshmen enrolled in the course (not just those with declared CS/SE/CE majors) was analyzed to see whether using pair programming made the general population any more likely to be CS/SE/CE majors one year later. The data was analyzed in the same three groups (all students, only females and only minorities). The results in Table 2 show that in all three cases, the students from the pair programming section were more likely to be CS/SE/CE majors one year later, but none of these results were significant.

Table 2 – Percent of students in majoring in CS/SE/CE 1 year after course

Variable	**Pair Programming**	**No Pair Programming**
% of CS/SE/CE majors 1 year later	85.4% (41/48)	64.3% (36/56)
Only females (CS/SE/CE)	66.7% (6/9)	33.3% (2/6)
Only minorities (CS/SE/CE)	75% (9/12)	70% (7/10)
% of all majors 1 year later	54.7% (41/75)	45.6% (36/79)
Only females (all majors)	54.5% (6/11)	28.6% (2/7)
Only minorities (all majors)	56.3% (9/16)	38.9% (7/18)

4.2 Post-hoc Analysis

To more fully understand the result described in Section 4.1 and better understand whether the use of pair programming contributed to the increased retention, the answers the students gave to surveys conducted throughout the semester were analyzed. These surveys were given anonymously and were not required; therefore we were not able to limit this analysis to only the responses from freshmen subjects as was done for the retention analysis (Section 4.1). The assumption is that the overall trend of the responses still provides insight into how the use of pair programming was perceived by the students. So, this information is still quite valuable.

The goal of these surveys was to determine whether the students found pair programming useful and beneficial and whether they believed it helped them in the course. The specific survey questions are shown in Section 3.2. Data was collected from the following surveys:

- *Partner Evaluation Survey (2)* – completed at the end of pairing 1 and the end of pairing 2 (Section 4.2.1);
- *Post-lab Survey (3)* – completed after lab assignment 1, lab assignment 2, and lab assignment 6 (Section 4.2.2).

4.2.1 Partner Evaluation Data: The first two questions on the *Partner Evaluation Survey* provided insight into the mechanics of the pairing. The students were asked whether they perceived their skills to be better, worse or equal to those of their partner. Earlier research has suggested that partners with similar perceived skill levels are the most effective; therefore, we wanted to ensure that our mechanism for pairing up students was adhering to this principle. In the first survey, in which the students were paired randomly, the mean was 2.43 (lower than the neutral score of 3), indicating that overall students perceived their partners to be slightly weaker than themselves. For the second pairing, we attempted to pair each student with someone of similar skill (based on their exam grades up to that point). On the second survey, this mean improved to 2.74 indicating that the students overall believed their skills were more evenly matched with their partner. This increase was significant using a t-test (p = .07).

The next survey question asked the students to indicate how well they got along with their partner. A vast majority of the students either agreed or strongly agreed with this statement for a mean value of 4.44 out of 5. Using a One-Sample t-test, the mean score of 4.44 was significantly higher than a neutral response of 3 ($t_{184} = 24.977$, $p < .001$). This result (shown in Figure 1) strongly supports the conclusion that the students got along with their partners. The fact that the students got along well with their partners may be a contributing factor to the increased retention of those students.

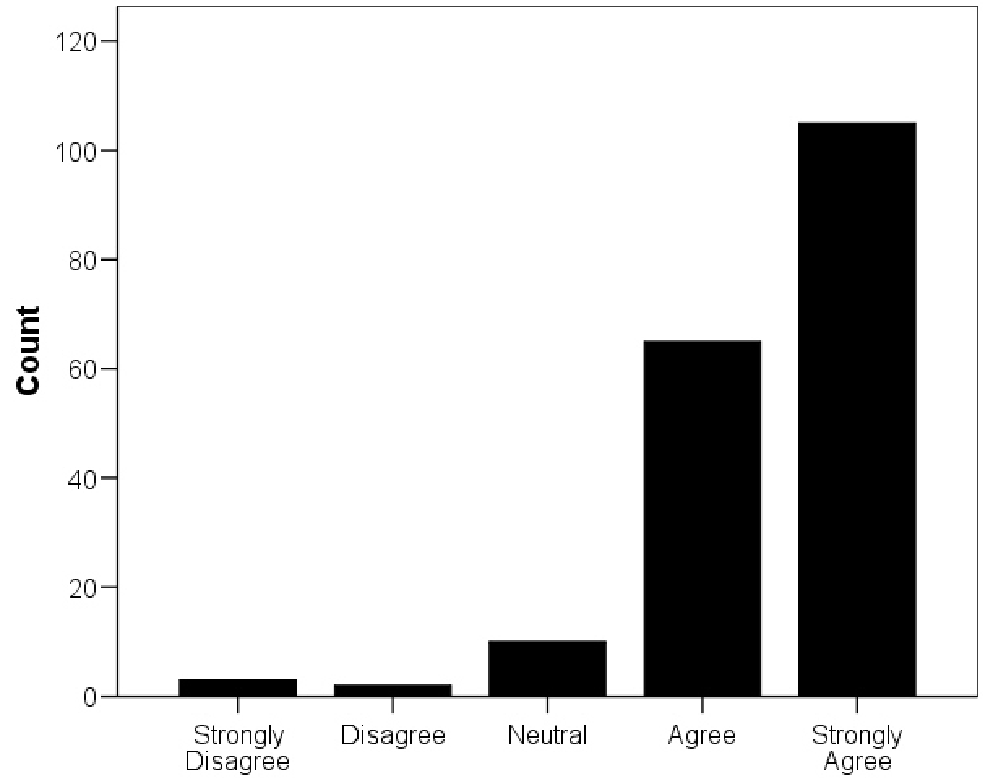

Figure 1 – Got Along with Partner

The third question posed to the students asked them to indicate whether they learned the concepts covered during the lecture portion of the course by working with a partner during the lab exercises. The average response to this question (3.37) was again significantly higher than the neutral score of 3 ($t_{184} = 3.797$, $p < .001$). This result indicates that one of the benefits of pair programming seen by the students was increased learning of the course material.

The last two questions asked the students whether they gained a better understanding of the course concepts by explaining them to their partner and whether they thought they were more efficient in debugging their code while working with a partner. The mean score for both of these questions was slightly higher than the neutral score of 3, 3.08 and 3.16 respectively; but these differences were not significant.

4.2.2 Post-Lab Survey Data: This survey allowed the students to evaluate the individual laboratory programming exercises. This information was collected after each of the first two assignments and at the end of the sixth assignment. In the analysis presented in this section, the responses from all three surveys are analyzed together for a total of 324 data points.

The first two questions on this survey were used to evaluate whether the students properly used the pair programming approach. The students were asked 1) whether they felt that the work was evenly divided between the partners and 2) if they took turns driving. For each question, the mean value, 3.81 and 3.75 respectively, was higher than the neutral score of 3. Each of these differences was significant based on the One-Sample t-test ($t_{323} = 12.169$, $p < .001$; $t_{323} = 11.829$, $p < .001$).

The third question asked the students whether they believed that they learned more by working on the laboratory exercise with a partner than they would have by working alone. The data in Figure 2 show that the students strongly agreed with this statement. Again the mean of 3.86 was significantly higher than the neutral score of 3 ($t_{323} = 12.008$, $p < .001$). This result indicates that another benefit of pair programming seen by the students was the ability to learn from their partner.

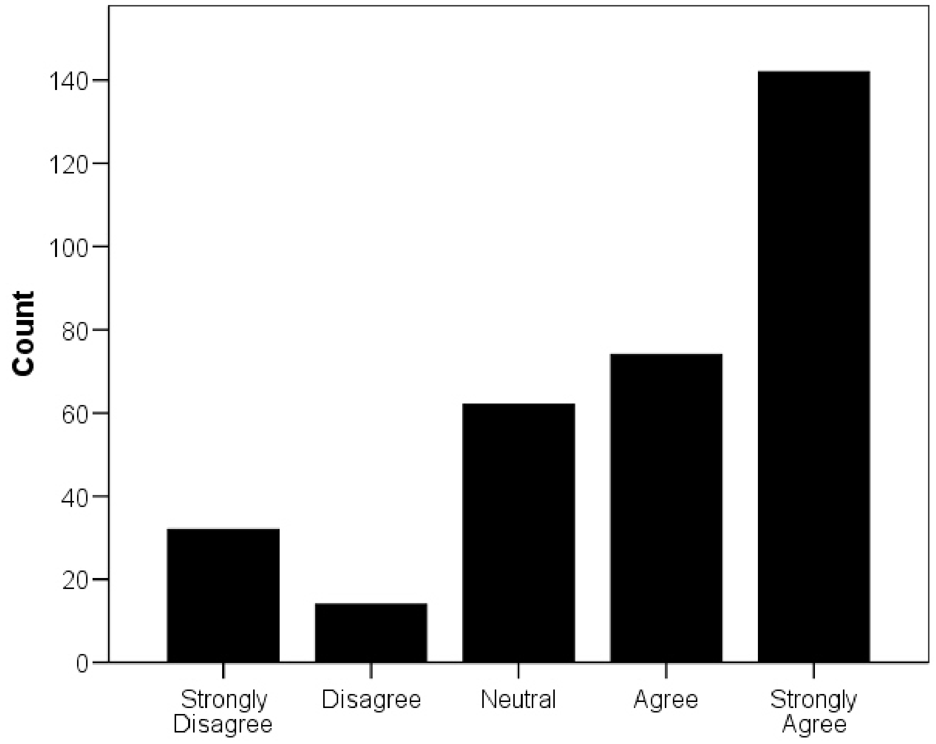

Figure 2 – Learned more by working by working with partner

5. Conclusion and Future Work

This paper described a study which was conducted to determine whether the use of pair programming for the laboratory exercises in the introductory programming course at Mississippi State University would help with retention of students. In addition to addressing this top-level question, we were also interested in understanding the reasons why pair programming might or might not be effective. This study showed that for students who enrolled in the introductory programming course with a declared major of CS/SE/CE, those who took the pair programming version of the course were significantly more likely to still be CS/SE/CE majors one year later than students who took the non-pair programming version of the course. In addition, when comparing the percentage of all students, regardless of their major at the time of the course, those who took the pair programming version of the course were also more likely, although not significantly, to be CS/SE/CE majors one year later. The results for females and minorities also showed that pair programming increased retention, but due to the small size of the subject population, no valid statistics could be computed.

The results also provided some insights into the potential reasons for increased retention. In terms of the logistics of using pair programming, the students strongly agreed that they got along well with their partner. They also rated their partner's skill level to be slightly weaker than their own. But this rating significantly increased when the pairings were done based on test scores. There was also strong evidence that the students believed that the workload was evenly divided and that they took turns driving during the laboratory exercises. From an educational point of view, the results indicated that the students got along well with their partners. In addition, the fact that the students strongly believed that they learned the course concepts well when working with a partner and that many of the students said they benefited from explaining the course materials to their partner indicate that the pairings were a positive aspect of the course.

Overall, these results are very encouraging both from a retention and from an educational point of view. Based on these results, we have continued to use pair programming in our introductory programming course and are expanding its use to use it in other courses. We plan to continue monitoring the retention and education of the students enrolled in these courses to continually improve our process. Another aspect of our future work will be to replace the Myers Briggs personality test with another one more accepted by psychologists to determine whether it has any impact on pair performance.

6. Acknowledgements

We would like to thank the students who participated in the study. We would also like to thank Dr. Edward Allen for reviewing an early draft of this paper.

7. References

[1] Beck, K., *Extreme Programming Explained: Embrace Change*. 2000, Reading, MA: Addison-Wesley.

[2] Hanks, B. "Student Attitudes toward Pair Programming". In *Proceedings of The 11th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE'06)*. Bologna, Italy: ACM. June 26-28, 2006. p. 113-117

[3] McDowell, C., Werner, L., Bullock, H.E., and Fernald, J. "The Effect of Pair Programming on Performance in an Introductory Programming Course". In *Proceedings of ACM Special Interest Group of Computer Science Educators*. Kentucky. 2002. p.

[4] McDowell, C., Werner, L., Bullock, H.E., and Fernald, J. "The impact of pair programming on student performance, perception, and persistence". In *Proceedings of 25th International Conference on Software Engineering*. Portland, OR. 2003. p. 602-607

[5] McDowell, C., Werner, L., Bullock, H.E., and Fernald, J., "Pair programming improves student retention, confidence, and program quality." *Communications of the ACM*, 2006. **49**(8): 90-95.

[6] Nagappan, N., Williams, L., Ferzli, M., Wiebe, E., Yang, K., Miller, C., and Balik, S. "Improving the CS1 Experience with Pair Programming". In *Proceedings of SIGCSE'03*. Reno, Nevada, USA: ACM. February 19-23, 2003. p.

[7] Werner, L.L., Hanks, B., and McDowell, C., "Pair-Programming Helps Female Computer Science Students." *ACM Journal of Educational Resources in Computing*, 2004. **4**(1).

[8] Williams, L., Kessler, R.R., Cunningham, W., and Jeffries, R., "Strengthening the Case for Pair-Programming." *IEEE Software*, 2000. **17**(4): 19-25.

[9] Williams, L., Wiebe, E., Yang, K., Ferzli, M., and Miller, C., "In Support of Pair Programming in the Introductory Computer Science Course." *Computer Science Education*, 2002. **12**(3): 197-212.

[10] Williams, L., McDowell, C., Nagappan, N., Fernald, J., and Werner, L. "Building Pair Programming Knowledge Through a Family of Experiments". In *Proceedings of International Symposium on Empirical Software Engineering* Rome, Italy. 2003. p.

[11] Williams, L., "Debunking the Nerd Stereotype with Pair Programming." *IEEE Computer*, 2006. **39**(5): 83-85.